NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

DESIGN AND IMPLEMENTATION OF VISUAL INTERFACE TO DATABASE

by

Suprapto

September 1991

Thesis Advisor: Dr. C. Thomas Wu

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution is unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: Computer Science Dept. Naval Postgraduate School; 6b. OFFICE SYMBOL (if applicable): CS	7a. NAME OF MONITORING ORGANIZATION: Naval Postgraduate School
6c. ADDRESS (City, State, and ZIP Code): Monterey, CA 93943-5000	7b. ADDRESS (City, State, and ZIP Code): Monterey, CA 93943-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION; 8b. OFFICE SYMBOL (if applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
DESIGN AND IMPLEMENTATION OF VISUAL INTERFACE TO DATABASE

12. PERSONAL AUTHOR(S)
Suprapto

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Master's Thesis	FROM 09/90 TO 09/91	September 1991	127

16. SUPPLEMENTARY NOTATION: The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Visual interface, Visual programming,Software Development Tools

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Traditional approaches to database system design and implementation involve text-oriented data access with their inherent lack of modularity, extensibility and modifiability. An alternative to this traditional approach is using visual interface for the design and implementation of databases. This alternative approach involves using software development tools(toolkits) to ensure modularity, extensibility and modifiability. To study the effectiveness of using visual interfaces, we have designed and implemented a sample application using KnowledgePro(Windows), a tool for rapid application development under Windows.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL: Dr. C. Thomas Wu	22b. TELEPHONE (Include Area Code): (408) 646-#### / 22c. OFFICE SYMBOL: CS/Wq

Approved for public release; distribution is unlimited

DESIGN AND IMPLEMENTATION OF VISUAL INTERFACE TO DATABASE

by
Suprapto
Captain, Indonesian Air Force
B.A., Surabaya Institute of Teachers and Educational Sciences, 1978
Drs. degree, Surabaya Institute of Teachers and Educational Sciences, 1981

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 1991

Author: _____
Suprapto

Approved By: _____
Dr. C. Thomas Wu, Thesis Advisor

_____
CDR Bruce B. Giannotti USN, Second Reader

_____
Dr. Robert B. McGhee, Chairman,
Department of Computer Science

ABSTRACT

Traditional approaches to database system design and implementation involve text-oriented data access with their inherent lack of modularity, extensibility and modifiability. An alternative to this traditional approach is using visual interface for the design and implementation of databases. This alternative approach involves using software development tools(toolkits) to ensure modularity, extensibility and modifiability. To study the effectiveness of using visual interfaces, we have designed and implemented a sample application using KnowledgePro(Windows), a tool for rapid application development under Windows.

TABLE OF CONTENTS

I. INTRODUCTION 1

A. MOTIVATION 1

B. THESIS ORGANIZATION 2

II. BACKGROUND 4

A. VISUAL INTERFACE 4

1. Visual Programming 6

2. The Categories of Visual Programming 7

a. Languages for handling Visual Information 8

b. Languages for supporting Visual Interaction 8

c. Languages for programming with visual expressions 9

d. Visualization of Data or Information about Data 10

e. Visualization of Program/ Execution 10

f. Visualization of Software Design 10

g. Visual Coaching 11

B. WINDOW INTERFACE 12

1. Window Manager 12

2. Window Systems 14

C. KNOWLEDGEPRO(WINDOWS) 15

1. Getting Started 15

2. User Interaction 17

3. Some Tools Provided 18

III. DESIGN AND IMPLEMENTATION 20

A. PROBLEM STATEMENT 20

1. The Current System 20

2. The Proposed System 20

3. The Design Methodology 21

4. Design Considerations 22

B. THE GOAL OF IMPLEMENTATION 22

C. THE EMPHASIS AREA INFORMATION SYSTEM 22

1. The Object of EAIS 22

2. Application(functional) Requirements 25

3. Internal Processes 27

4. The Query Database Module 29

5. Database Management Module 36

6. Coding the Programs 40

IV. EVALUATION 43

A. THE VISUAL INTERFACE CATEGORY OF KNOWLEDGEPRO . . 43

1. Linking screen objects to topics 44

2. Displaying a picture 45

3. Handling events 46

4. Getting information from the user 47

5. Temporarily halting topic execution 48

B. THE ADVANTAGES AND DISADVANTAGES OF USING KNOWLEDGEPRO 49

1. Advantages 49

a. Ease of Learning 50

b. Ease of Use 50

c. Modularity 52

2. Disadvantages 53

a. Speed 53

b. Error Handling and Compiling 53

c. Design Tools lacking 54

C. COMPARATIVE STUDY BETWEEN TOOLBOOK AND KNOWLEDGEPRO 54

1. ToolBook 55

2. KnowledgePro 55

V. CONCLUSIONS 57

APPENDIX A. SOURCE CODE LISTING 59

LIST OF REFERENCES 116

INITIAL DISTRIBUTION LIST 118

I. INTRODUCTION

A. MOTIVATION

Recent advances in science and technology have made high-resolution graphics an available presentation medium on mainframes and personal computers. Tremendous advances in computer hardware have also contributed to the demand for software applications which take advantage of these hardware capabilities. As a result of these changes is the flourishing of database applications employing visual interactions as means of communication between users and their data base systems.

Almost all traditional programming languages developed to date have a text-based syntax and structured modularized software development approach. Creating a graphical display in traditional approach is to write program *which* accepts parametized input, examine a database, and then make calls to a graphic subroutine package to create the desired display. This approach involves using relatively low level procedural languages such as Ada, or C. The complex nature of Visual Interface software requires unacceptable amounts of time to design, code, test and maintain the software.

By using a relatively a new approach, this time problem can be overcome. This new approach involves using Software Development Tools (toolkits) to develop Visual Interface software. Toolkits should be designed to ensure reusability, modifiability and extensibility. That also means, the tools should help ensure consistency, ease of maintenance, and expedient implementation. These characteristics should be used as

criteria for selecting a toolkit. Typically, these toolkits provide a collection of standard objects that should be powerful enough to be used to build visual interfaces with small amounts of code. For these reason, it is necessary for database managers to find a toolkit that support the desired characteristic.

Many different Visual Interface Development Tools have been proposed in recent years. One of the promising toolkit that supports the desired characteristics is KnowledgePro (Windows). KnowledgePro (Windows) contains high-level commands for manipulating screen objects. It also utilizes direct manipulation of objects in the database, which extremely facilitates using and learning the system.

The goal of this thesis is to study the ease and the effectiveness of visual interface development tools. To achieve the goal, we demonstrate the design and implementation of prototype application Emphasis Area Information System (EAIS) using toolkit KnowledgePro(Windows). This system includes the required and optional electives for each emphasis area supported by the database system which enables Computer Science personnel to select elective from an interactive scheduling program.

B. THESIS ORGANIZATION

Chapter II will address some background information for the thesis, beginning with a discussion of visual interface. Discussion will include visual programming, the categories of visual programming, the advantages and disadvantages of each category in general. We then present the features of window interface and, finally, the overview of KnowledgePro(Windows).

Chapter III will include a discussion of a prototype implementation created with KnowledgePro (Windows). The discussion will describe the problem statement, the goals of design and implementation, the Emphasis Area Information System, using the proposed software KnowledgePro (Windows).

Chapter IV will evaluate the application development process. We then discuss the category of KnowledgePro (Windows), its advantages and disadvantages compared to other Visual Interface Development Tools Toolbook. Finally, chapter V will present the conclusion of the research.

II. BACKGROUND

In this chapter, we describe some background information of the thesis. Section A looks at the significance of visual interface for human being. Section B discusses the features of window interface. The final discussion is an overview of corresponding software development tools which is used in our research.

A. VISUAL INTERFACE

During the human-computer interaction, most information conveyed by computer is presented visually, typically by means of dynamic display device. Today, the majority of these devices are cathode ray tubes (CRTs), although other devices such large-scale liquid-crystal displays may be more common in the future. No matter what display technology is used, the content and format of information presented will always have substantial impact on the usefulness and efficiency of the interaction. In the human-computer interaction, visual interface affects both the user initial impression of the interface and system's longer-term usefulness. Visual interface comprises all the graphic elements of interaction, including overall screen layout, menu and form design, use of color, information codings, and placement of individual units of information with respect to one another. Therefore, graphical displays play important role in visual interactions.

The trend in the design of a database systems is towards more visual forms of interaction. The means of communication is no longer simple comprehension of written words, but interpretation of visual and spatial properties of graphical items including

icons, windows and menus. We believe a visual interface holds a best potential as a end-user interaction tool.

In recent years, many visual interface to database systems have been proposed(Hero 1980, Lars 1984, Wu 1986, Wu 1987, Wu 1988) with different approaches and different designs. Good visual interface design strives for clarity, consistency, and attractive appearance(Foley, 1987). Wu and Hsiao (1988) have identified six principles for the development of visual interface design GLAD (Graphics LAnguage for Database). The program should be able to :

- Provide more information when asked.
- Recover from the unintended or erroneous operation.
- Perform the same operation in more than one way.
- Perform logically equivalent operations in a consistent manner.
- Display multiple information at the same time.
- Display multiple views of the dame information.

The important of visual interface design has been demonstrated in a variety of studies. For example, Tullis (1981) found that redesigning a key display from a system for testing telephone lines resulted in a 40% reduction in the time required by the user to interpret the display. Wu and Hsiao (1988) found that designing GLAD interface utilizes direct manipulation of objects in the database through the use of buttons and other controls which make using and learning the system extremely easy. Likewise, Keister and Gallaway (1983) found that redesigning a series of screens resulted in a 25% reduction

in the total processing time and a 25% reduction in error rates.

Visual information such as graphical representation and pictures are more powerful than words as means of communication. Graphics and pictures also aid understanding and remembering. In this section we discuss visual programming, the categories of visual programming and its advantages and limitations each category in general.

1. Visual Programming

In the early days of the interactive computing when the hardware and software were limited, system designers focused on processor efficiency, commonly more at the expense of the user than on the human convenience. Additionally, as we stated in chapter I, almost all traditional languages were not designed for visual interactions.

However, as hardware costs dropped and computer systems became more widespread, data set sizes grow, algorithms become more complex, the cost of user's time greatly increase. The price of RAM chips fell enough to accommodate the huge memory demands of bit-mapped graphics. These hardware and personnel costs, coupled with competition in the marketplace, and the need of visualization systems, have caused designer to focus on human-computer interface. This situation gives challenge to designer to bring computer capabilities more powerful than currently available, usefully and simply to the user without special training.

As a result, in recent years, the use of visual information has gained momentum in programming. This kind of programming, called visual programming, is done through visual interaction with the system. Visual programming provides more direct communication between a person and a machine and makes it easier to create programs.

Visual programming makes greater use of extensive visual information processing capabilities of human, and by doing so, provide for totally new types of human-computer interaction(Tullis, 1988).

2. The Categories of Visual Programming

Programming can be defined as specifying a method for doing something the computer can do, especially in terms the computer can interpret. To attempt the ease of programming, of course, is not new. People have been trying to design or improve programming language for the ease of use and the ease of learning for several years. However, language design has been evolutionary rather than revolutionary. Since programming is currently to be made more accessible to people, it become apparent that radical departure from traditional programming into visual programming.

They are several aspects of programming : the languages and environment used for the specifications; the specifications themselves; the display of data involved in the execution of the specifications; the determination of whether the computer has executed a specification as expected; etc. All aspects of programming can be applied by visual programming. To determine whether a specific software is visual programming or not, is the consideration of some meaningful graphical representations must be used in the programming process.

From the recent work reported in the literature, we see that visual programming progress in two directions. In one direction, languages are designed for handling visual information; for supporting visual interactions; and for programming with visual expressions. In another direction, pointing devices and graphical representations are

used to provide visual environments for program development and execution; for information retrieval and presentation; and for software design and understanding(Nan C. Shu, 1988).

a. Languages for handling Visual Information

This language is designed to process image or pictorial information. The image processing systems designed for handling of two-dimensional pictorial data that can be applied for specific purpose such as an engineering, a geographical, a medical and a scientific applications. Now, incorporation between conventional database query languages and image processing systems have already be implemented as augmented conventional query languages. Languages fall into this category are GRAIN(the Graphic-oriented Relational Algebraic INterpreter), QPE(Query-by-Pictorial-Example), QBE(Query-by-Example), PSQL(Pictorial Structured Query Language).

The benefits of the system that it can store and retrieve pictorial information into database. The system also includes zooming technique, such as vertical zoom, horizontal zoom and diagonal zoom, makes the system extremely attractive. The complexity increases when we have a large number of pictures or very large pictures.

b. Languages for supporting Visual Interaction

Languages in this category are mainly designed to define, create, and manipulate pictorial symbols. It is also called Iconic Programming. They are motivated by the need to have the ease of using and manipulating of pictorial data. The languages that fall in this category, generally support visual representations and visual interactions.

Although the languages themselves are actually textual, not visual. Languages fall into this category are ICDL(icon-class description language), HI-VISUAL, and Squeak. ICDL is language that consists of several statements. Each statement accepts some value and performs some graphical operation. HI-VISUAL is a language supporting visual interaction in programming. Squeak is a language supporting communication with a pointing device.

The advantage of this approach over textual programming is that it can make learning and understanding easier, especially for non-computer specialists. Iconic is effective for introductory courses in languages. Hirakawa, Tanaka, Ichikawa(1984) proved through an experiment that iconic languages are easier to learn than textual one. Iconic programming may be good for small programs. In contrast, iconic programming is the wasteful use of screen space. It make difficult to make a large program.

c. Languages for programming with visual expressions

Languages that fall into this category, focus on allowing users to program with visual expressions. These languages are usually called the "visual programming languages". They use some visual representations(textual, numeric, pictorial or audio) as means of programming. Almost all visual programming languages reported from literature fall into three categories. First category is Iconic languages. In Iconic languages, icons and graphical symbols are designed to play the central role in programming. Second, Charts and Diagrams are either incorporated into programming development, or made into machine-interpretable units to be used in conjunction with traditional programming languages. Third, Form-based languages lie between chart/diagrams and iconic languages.

d. Visualization of Data or Information about Data

This approach typically concentrates in the environment area which deals with visualization of data or information about data. The data are kept in traditional databases. It expressed in graphical form and presented to the user in a spatial framework. Tools are provided which allow the user to make connection between data from these sources and their associated graphical representations. By using pointing device or joystick the user directly manipulates data in the databases. The examples of this category are SDMS(Spatial Data Management System), VGQF(Video Graphic Query Facility), INCENSE(a System for Displaying Data Structures) etc.

e. Visualization of Program/ Execution

Languages in this category primarily designed to provide graphical representation for the visualization of programs, run time states and results. The goal is to use graphical displays to make the program development and testing easier. This is benefits not only to beginners learning how to program, but also to experience programmer dealing with program development. Activities in this area span from "pretty-printing" the source code, watching the execution program in multiple displays or in animated forms.

f. Visualization of Software Design

There are many activities in the software development process. The activities include requirement analysis, functional specifications, design decisions, implementation, system structures. Languages that fall into this category provide graphical

forms in the software development environment. People who design, use, or maintain interact graphically with the system. The examples of this category are PeaSys(Programming Environment for the Graphical Analysis of SYStems), PV (Program Visualization) system.

g. Visual Coaching

Visual Coaching actually attempts to produce program based on what has been shown. It can be characterized as "Do what I show you" or "Do according to my signal." Typically, the users show the computer how to do the desired computation by demonstrating with some sample data. The system then applies the demonstrated behavior for application with other data. Program development completely depends on interactions. Languages that fall into this category are Programming By Example, Programming By Rehearsal, AutoProgramming.

There are several advantages using this approach. This approach attempts to extend ideas to abstract concepts in programming. Because most people are much better dealing with concrete and specific objects than with abstract ideas. So, by demonstration and examples can help people to understand abstract concepts. The users find it easily to demonstrate what they want done. Additionally, this approach can assist the thinking and learning process of human beings. This approach relies on graphical interactions, so that program construction can be easy, quick, and enjoyable. On the other hand, this approach has limitations. First, physical limitations imposed by the space on the screen. Second, the difficulties to express an ideas in more abstract and general terms.

B. WINDOW INTERFACE

Windowing interfaces got their start at the XEROX Palo Alto Research Center (PARC) in the mid-1970's. In 1984, they were popularized by Apple Computers, particularly the Macintosh line, and later, followed by Microsoft and other developers. Window interfaces has become popular primarily because they allow separate activities to be put in physically separate parts of the computer screen. This physical separation is even more important when the operating system allow multiple activities to operate at the same time ("multiprocessing"). Window interface provides many of the important features of modern user-computer interfaces applications that show results in different areas of the display, the ability to resize the screen areas in which those applications are executing, pop-up and pull-down menus, and dialogue boxes, the ability of repositioned windows. Now, many windowing interfaces offer command-key or keyboard alternatives for many commands as well, providing both ease of use for novices and the less experienced, and speed for who can type faster than they point to menu and click on a selection.

Window interface has two important parts. The first is the window manager, which allows the user to interact with computer, request that windows be created, resized, repositioned, opened, and so on. The second is the underlying functional component, the window system, which actually causes windows to be created, resized, moved, and so on.

1. Window Manager

The window manager uses user-interface services provided by the window system, promotes consistency and makes application easier. This includes all aspects that are visible to user such as a graphic package, editors, users interface tool kits, typescript

package, etc. The window manager consists into two parts : the presentation, which is composed of the pictures that the window manager displays, and the operations, which are the commands the user can give to manipulate the window and their contents. For example windows that demonstrate different aspects of the presentation, including patterns or pictures for the area where there are no windows, title lines and border for windows, etc may be provided the operations for windows include moving them around on the screen and specifying their size.

One important aspect of the presentation of the window is whether they can overlap or not. This also some time called desktop metaphor, since windows can cover each other like pieces of paper can cover on the desktop. The other alternative is called tiled windows, which means that windows are not allow to cover each other. Some researchers still question whether overlapping windows offer more benefits than tiled, at least above a certain screen size, on the grounds that screens with overlapping windows become so messy the user gets lost. Others argue that overlapping windows more closely match user's work patterns, since no one arranges the papers their physical desktop in neat horizontal and vertical rows. Among software engineers, however, overlapping windows seem to have won for the user interface world(Tekla s. Perry, and J., Voelcker 1989). Another important aspect of the presentation of the window is the use of icons. These are small pictures that represent windows, to manage too many windows conveniently.

2. Window Systems

Window systems implement the basic functionality of the window interface. They handle the display of graphics in windows and access to the various input devices (usually a keyboard and a pointing device such as a mouse). The primary interface of window systems is to other programs, and it is called the manager's application or program interface. The programs built on the window systems are usually called *client program*, which in turn use the capabilities of the window system, itself usually called the *server* program. A graphics package is often integrated with the window system. Figure 2.1 shows how the window system and its graphics package typically relate to other system component.

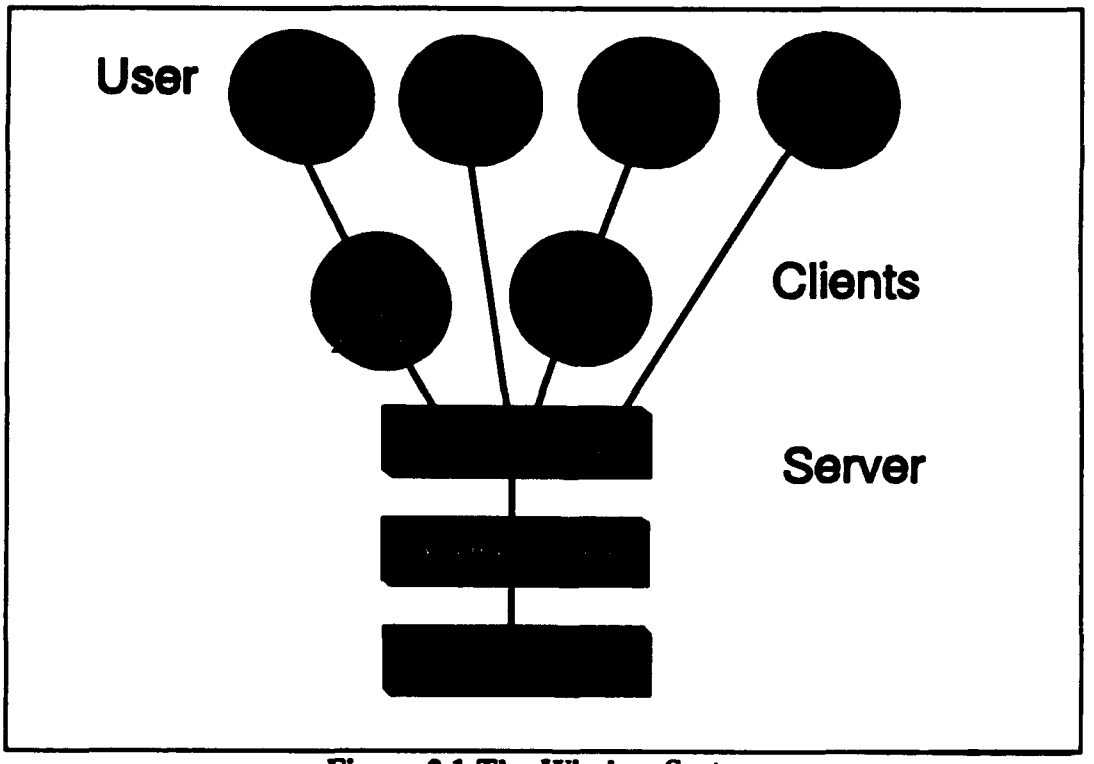

Figure 2.1 The Window System

C. KNOWLEDGEPRO(WINDOWS)

Today, several application development tools for Windows 3.0 are available in the marketplace. Some products, like Asymetrix ToolBook offer a development environment for sophisticated end-users. Others, like Whitewater Group's Actor are designed for programmers. One that fulfills the needs of both groups is KnowledgePro(Windows).

KnowledgePro(Windows) combines several significant new technologies in a remarkably accessible toolbox:

- Windows development capabilities
- hypertext and hypermedia access
- expert system strength

This technology makes it poswble to create Windows applications that combine the ability to link among various elements of a collection of information, to enhance the end-user interface. It also makes it possible to build Windows programs that can exhibi: behavior that would probably agree demonstrates at least rudimentary intelligence. (Dan Shafer, 1989)

The following, we discuss how to start KnowledgePro, user interacts with computer, and some tools provide ! by KnowledgePro.

1. Getting Started

Figure 2.2 shows the development environment running a simple application. KnowledgePro(Windows) includes an integrated, multiple-document editor for easy manipulation of source code and text. We can run quickly just by selecting Go from the menu. Whenever an error occurs, we are return to edit window at the line which caused

the problem. KnowledgePro(Windows) provides a full set of debugging tools, including Trace, Single-step and the ability to interact with running application.

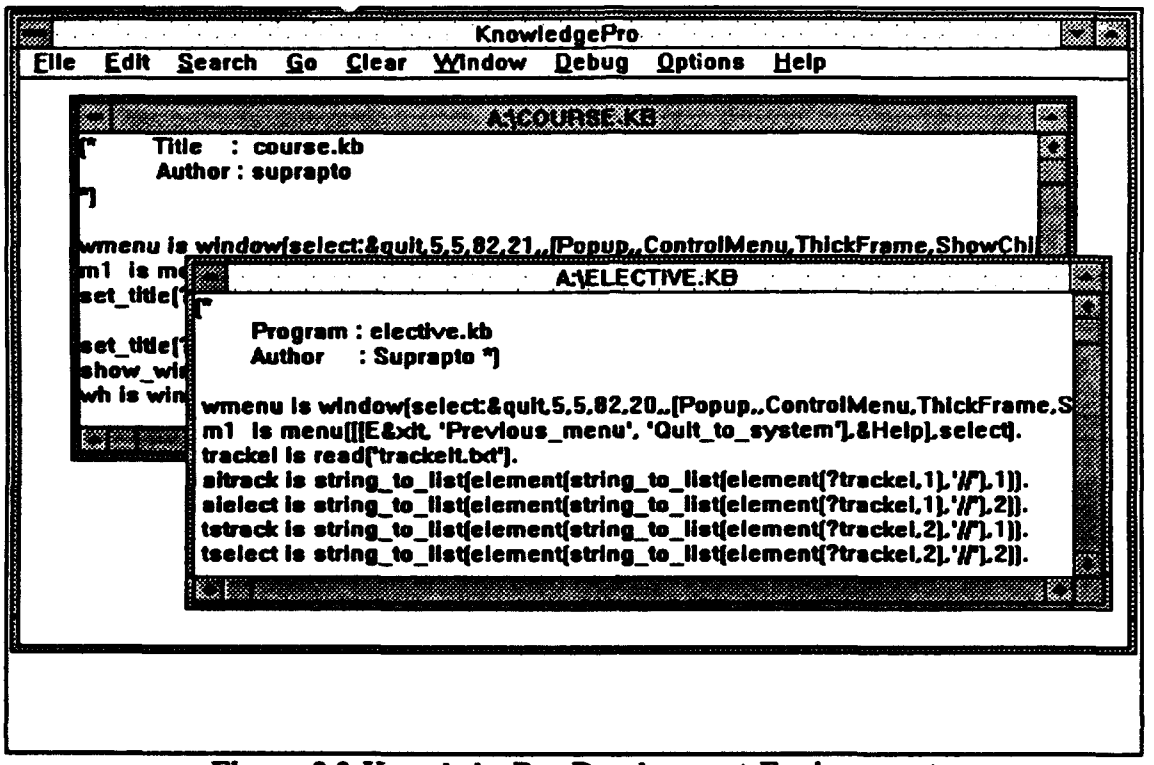

Figure 2.2 KnowledgePro Development Environment

The backbone of KnowledgePro(Windows) is KnowledgePro language: an event-driven, message-based language with object oriented features like multiple inheritance. Language, editor and tools provided by Knowledge processing environment, let the user communicate expertise using the computer. The program that the user creates in this environment is called Knowledge base. The example of Knowledge base is shown below:

window().

```
contents is '#e #mIntroduction#m
        #mMotivation#m
        #mThesis Organization#m'.
text(?contents).
topic mark(item).
    message is read('course.txt', concat('//'?item),'//').
    close('course.txt').
    say(?message).
    text(?contents).
end.
```

Each Knowledge base consists of one or more topics that contain commands, functions, and subtopics. These topics perform actions, prescribe a behavior, or service hypertext requests. Topics specify what happens when the user clicks on or selects an object. Once the user links topic to screen object, the object will act according to its defined behavior. A special topic named *mark* can be defined as a "catch all" topic for a group of hypertext topic that behave in a similar manner. This means that if a topic with the same name as the hypertext selected can not be found, the topic named *mark* is called in its place. If *mark* is found, the hypertext phrase selected is passed to it as a parameter, otherwise a message will appear that says no more information is available.

2. User Interaction

One of the advantages of using a computer is that we can gather information from the user. Through the knowledge base, the designer communicates to the user, and,

can also provide a means for the user to respond. Like in real conversation, this response influences the subsequent direction of the exchange. KnowledgePro let users use many different screen objects such as buttons, check boxes, list boxes, radio buttons, edit lines, edit boxes, scroll bar etc., that can be used to accomplished a desired conversation. In other word, KnowledgePro includes the ability to create every kind of control available in the Windows environment. For example, edit lines are used for getting values from the user, and created using **edit_line** command. KnowledgePro provides many different types of screen objects. Each one has its own characteristics making it suitable for certain types of information.

Any word, phrase or screen object can be linked to a KnowledgePro topic. This lets us build hypertext application with flexibility and power. By knowing which screen objects to select for which operations, how screen objects operate, how to arrange the objects in uncluttered, how to make visually pleasing display, we can create an effective visual interface application.

3. Some Tools Provided

Effective user interface and the communication of knowledge are becoming more and more important as computers, applications and users needs all increase in complexity. KnowledgePro(Windows) is provided by tools that lets the designer and the user communicate with each other about the user's needs, the resources available and the task at hand as conversation were taking place. Some tools provided by KnowledgePro Windows are the following:

- The KnowledgePro Editor that lets the users arrange edit windows, edit text, make backup, compile and execute an open Knowledge base file.

- Debugger that lets the users trace the program execution, list topics calls in a knowledge base, evaluate a knowledge base, inspect the running knowledge base with debug topics.

- Dynamic Data Exchange(DDE) for communicating with another window application.

- Design Tools that lets the users create, move and resize windows object in a window, generate the KnowledgePro code and copy to their application.

- Dynamic Link Libraries(DLL) that lets the users run external DOS program and extend the environment.

III. DESIGN AND IMPLEMENTATION

A. PROBLEM STATEMENT

Traditional approaches to database-system design and implementation involve text-oriented data access with their inherent lack of flexibility and extensibility. An alternative to the traditional approach to database-system design and implementation is visual interfaces. How does visual interface software effectively support data manipulation in a database system ? What are the benefits of using visual interface compared to regular text-oriented access ? To answer these questions we have developed prototype application Emphasis Area Information System(EAIS) using visual interface software.

1. The Current System

The selection of electives by students in the computer science departement is a manual process. Currently, a student receives a curriculum package which includes a schedule of the required courses for the final five quarters. A short description of the four emphasis areas available along with the required and the optional electives necessary to fulfill the requirements for the respective emphasis areas. When the electives are offered, the student fulfill the requirements of at least one emphasis area. The current system involves excessive administrative effort on the part of the curriculum staff and the student.

2. The Proposed System

To provide a automated database system which will consist of the required and optional electives for each emphasis area enabling Computer Science personnel to

select electives from this interactive scheduling program. This system will support Computer Science students; the Academic Associate; the Curriculum Officer as well as the Registrar. It will enable the user to view the curriculum requirements, and the elective requirements to supports various emphasis areas.

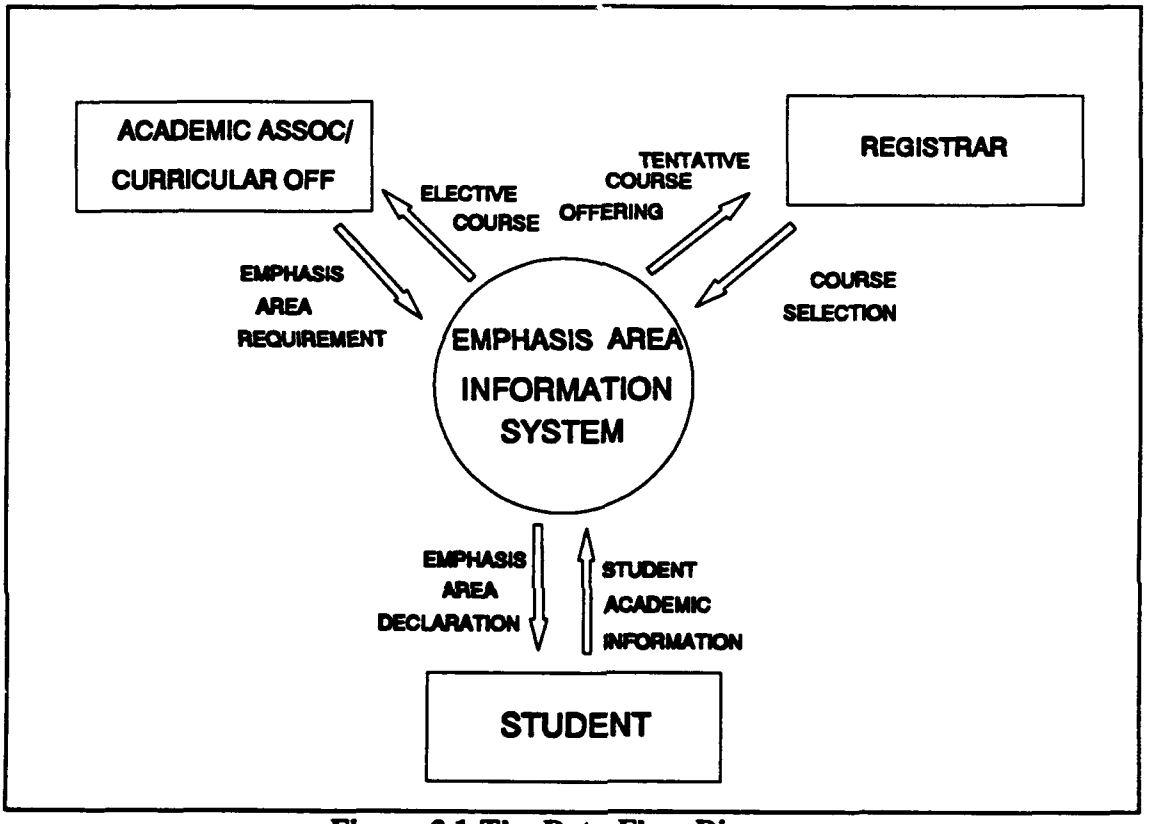

Figure 3.1 The Data Flow Diagram

3. The Design Methodology

The first step in designing a visual interface is to decide what the interface is meant to accomplish. We use a prototyping approach for developing and implementing EAIS. In this approach, we initially develop a conceptual design for implementation of EAIS. Figure 3.1 illustrates the Data Flow Diagram of our system. We then work on

elements of the functional design and dialog style. Finally, the prototyping tool software (KnowledgePro) can be used to develop and create screens or reports according to a dialog style prototype. The reason of using the prototyping approach in our system is to emphasize on speedy implementation and speedy modifiability.

4. Design Considerations

In our system, we apply a number of design principles to help ensure good human factors engineering in a design. They are to be consistent, provide feedback, minimize error possibilities, provide error recovery, accommodate multiple skill level, and minimize memorization. These and other principles are discussed more fully in (Morland 1983, Foley 1987, Shneiderman 1987, Mayhew 1990, Wu 1988).

B. THE GOAL OF IMPLEMENTATION

To study the effectiveness of developing visual user interface application using software development tools, we present the experience of developing a simple visual interface database application using toolkit KnowledgePro(Windows). The goal of the implementation is to show how easy and what are the benefits of using KnowledgePro (Windows) to develop visual user interface of Emphasis Are Elective Information System.

C. THE EMPHASIS AREA INFORMATION SYSTEM

1. The Object of EAIS

There are five objects in our system, *Course*, *Quarter*, *Emphasis*, *Elective* and *Student*. The relationship of the objects of the system is shown in Figure 3.2.

a. Course Object

The *Course Object* is established by Registrar. The properties of the Course Object consists Course Number, Course name, time slot for lecture and laboratory and course description.

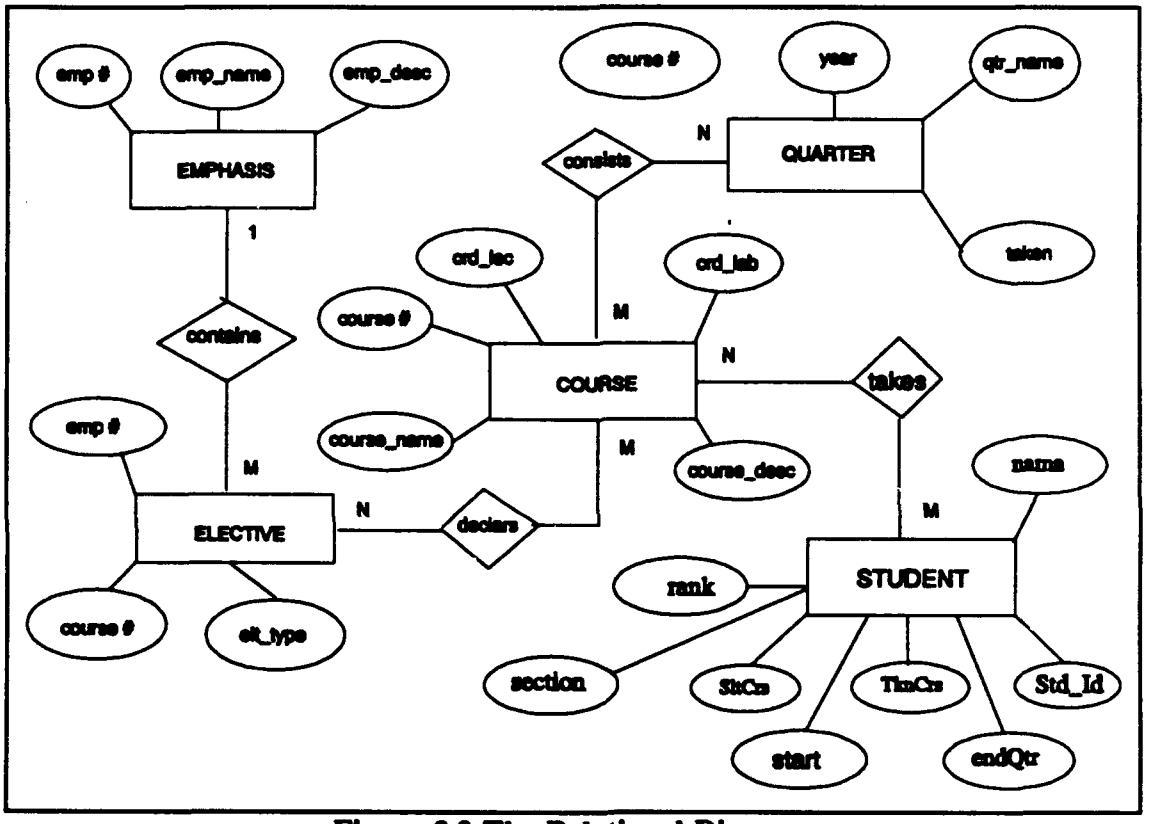

Figure 3.2 The Relational Diagram

b. Emphasis Object

The *Emphasis Object* is established by the Academic Associate and various sponsor, to concentrate on specific area within Computer Science. The properties of the Emphasis Object include the name of the emphasis area, a description of the

purpose of the emphasis area and a listing of elective courses that will fulfill the emphasis area. The elective course consists of the *Required Elective* and *Optional Elective.*

c. Elective Object

This *Elective Object* is specified by the Curriculum sponsor via Academic Associate and is outlined in the Naval Postgraduate School Course Catalog 1991. The properties of the Elective Object consists of the required course number, emphasis number and the elective type.

d. Quarter Object

The Registrar provide the Tentative Course Offering booklet which contains information about the expected course offering for the next year at Naval Postgraduate School. The *Quarter Object* is the subset of that listing which constains the courses that fulfill the requirements of the various emphasis areas extracted from the Registrar's Tentative Course Offering booklet. This properties of the Quarter Offering object includes the quarter name, course number, course name , when the course is projected to be offered in the next academic year.

e. Student Object

The *Student Object* contains a student academic information. This information is provided by the student. The properties of Student Academic Information include student's name, rank, section, selected emphasis area, selected courses, taken courses.

2. Application(functional) Requirements

The EAIS is supported by the database which consists of the objects described in the preceeding section. The system also interfaces with the external entities(terminators) in reaching decision on which the electives being requested will meet both the emphasis area requirements and the course scheduling constraints. The terminators that interface with the system include the Academic Associate, the Curricular Officer, the Registrar and the student user. The interfaces between the various terminators and the system are detailed below :

a. Academic Associate

The Academic Associate is responsible for :

- representing the interests of the sponsoring activities for the curriculum.
- providing the required courses for successful completion of the curriculum.
- determining the requirements for the four emphasis areas and providing a list of required and optional electives that will fulfill them.

The course requirements data provided by the Academic Associate are entered into a database and accessed by the system to support end user requirements. Modification of either the curriculum requirements or the Emphasis Area requirements will be conducted by the Curricular Officer at the direction of Academic Associate. Academic Associate receives the final report from the system and validates the request prior to its entry into the curriculum database.

b. The Registrar

The Registrar provides the Tentative Course Offering updated per

quarterly for Naval Postgraduate School from which the subset Emphasis Area Course Offering Object is extracted. The Emphasis Area Course Offering will be created and entered into database by Curricular Officer. The Registrar receives final report as soon as it has been entered into the curriculum database.

c. The Curricular Officer

The Curricular Officer will update Emphasis Area Course Offering Object as Registrar modifies the annual course offering listing for Naval Postgraduate School. Additionally, he modifies the Emphasis Area Object and the schedule as directed by the Academic Associate. The Curricular Officer receives the final report and has it entered into the curriculum database.

d. The Student

The student is primary interactive user of the system. The student enters initial data for the final report (Student name, rank and section), and the interacts with the system to complete the final report. The Student selects an emphasis area and proceeds to select the quarter he is interested in. The system will generate a listing of the Computer Science core courses that the student will be required to take during that quarter. The system also presents a listing of all elective that are contained in the specified emphasis area that are offered during the quarter that is being viewed.

The student has the option to view additional information about :

- the elective he is interested in, or
- he may enter the elective into a tentative final report format, or
- he may select a different quarter/ emphasis area for viewing.

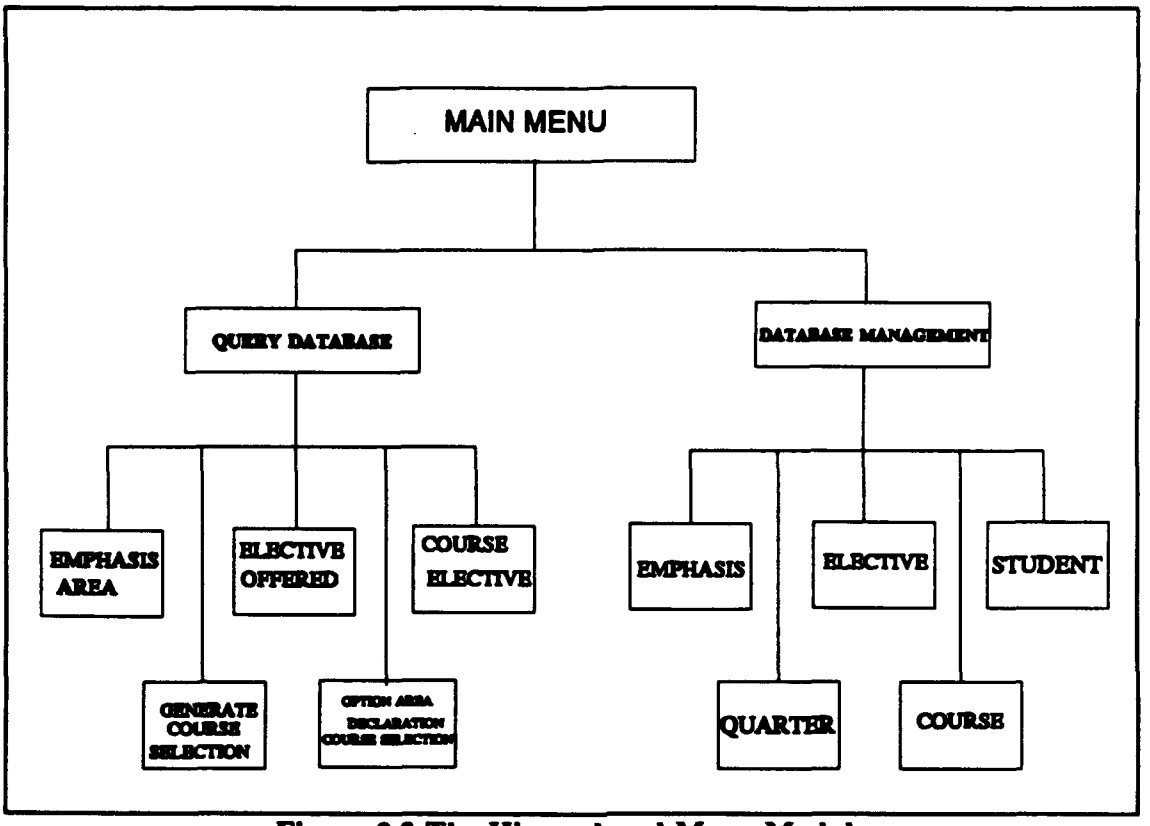

Figure 3.3 The Hierarchycal Menu Module

Once the student has created a tentative final report he has the option of re-entering the system to modify the report or printing the final report. Whenever the final report is created, the student signs the report and forwards it via the Academic Associate to the Curricular Officer and it is eventually entered into Curriculum database.

3. Internal Processes

The internal processes for the system include initial data entry and updating mechanisms for the database in the system. In the previous section we see that EAIS has five data object. Every data object is stored as text file namely *Emphasis.txt*, *Quarter.txt*,

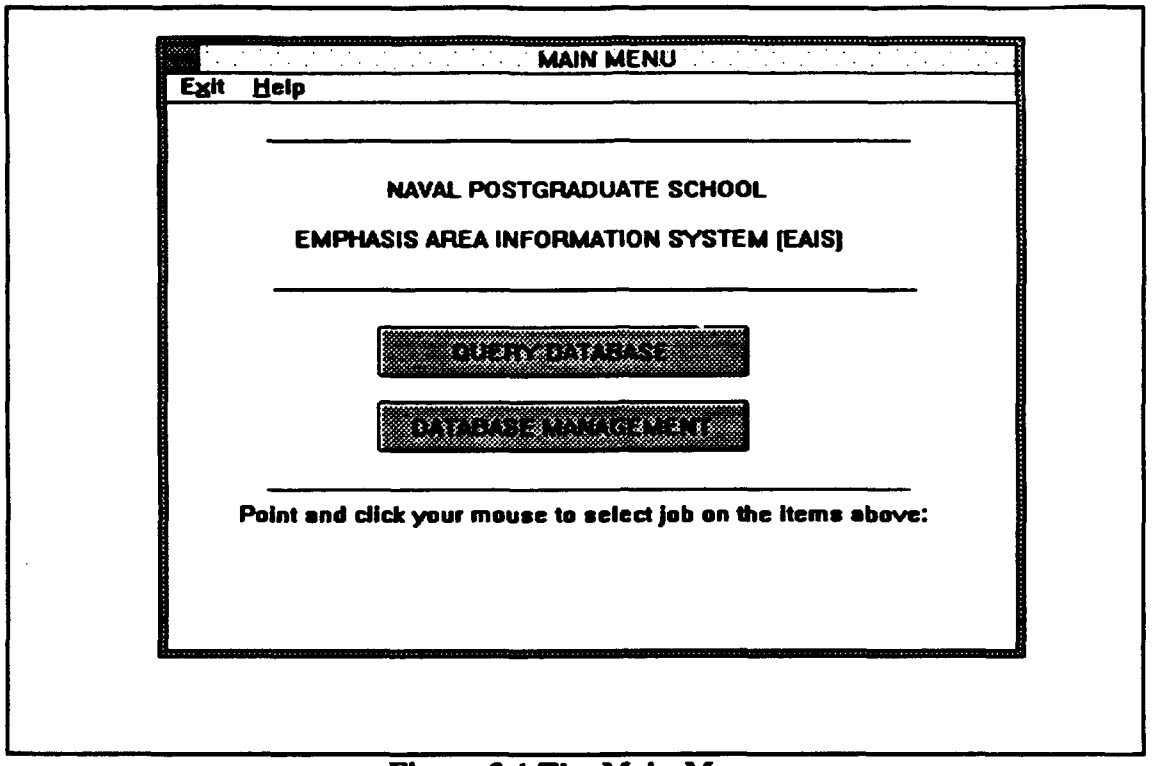

Figure 3.4 The Main Menu

Elective.txt, *Course.txt* and *Student.txt*. Because KnowledgePro only supports text file. To perform updating and querying a database, we implemented the list structure supported by KnowledgePro. These mechanisms can be done by first reading the data into lists. We then manipulate the data using list operations to get the desired result.

In our system, we use the action/object strategy in structuring hierarchy (see figure-3.3). With the action/object strategy, the highest level menu (main menu) begins with a list of action menus. When the user selects on an action, lower level menus will pick up an object on which to perform the action. Menu selections are attractive because they can eliminate training and memorization of complex command sequence. We wrote the menu items using familiar terminology, so it helps the user select an item easily.

Menus serve the purpose of showing the options the user can choose.

The main menu (figure 3.4) consists of the *Exit* and *Help* pull-down menu options, the *Query Database* option and the *Database Management* option. The *Exit* option ends the main menu session. We use three different methods to end the session. Go to the previous session, first we select the *Exit* option and we then select the *Previous_Menu* option. To exit the system, first we select the *Close* option from the control menu. Secondly, by selecting the *Quit_the_system* option from the Exit pull-down menu option. This means that the system provides a way to accommodate user of multiple skill levels. The *Help* option provides information to the user about the main menu choices. During human-computer interaction, the *Help* option can minimize error possibilities. When the user selects the *Query Database* option, the Query Database module promptly appears on the screen. This module is used for querying the database and generating the reports (see figure 3.5). Likewise, the Database Management module will appear on the screen as soon as the user selects the Management Database option (figure 3.14).

4. The Query Database Module

The **Query Database** module consists of button options. There are the *Emphasis_Areas*, the *Elective Offered*, the *Course Elective*, the *Option Area and Course Selection* and the *Generate Courses selection* options.

a. Query Emphasis Areas

This module is for querying the Emphasis area. To accomplish this job, we first select the *Emphasis_Area* pull-down menu option. We then select one of the track

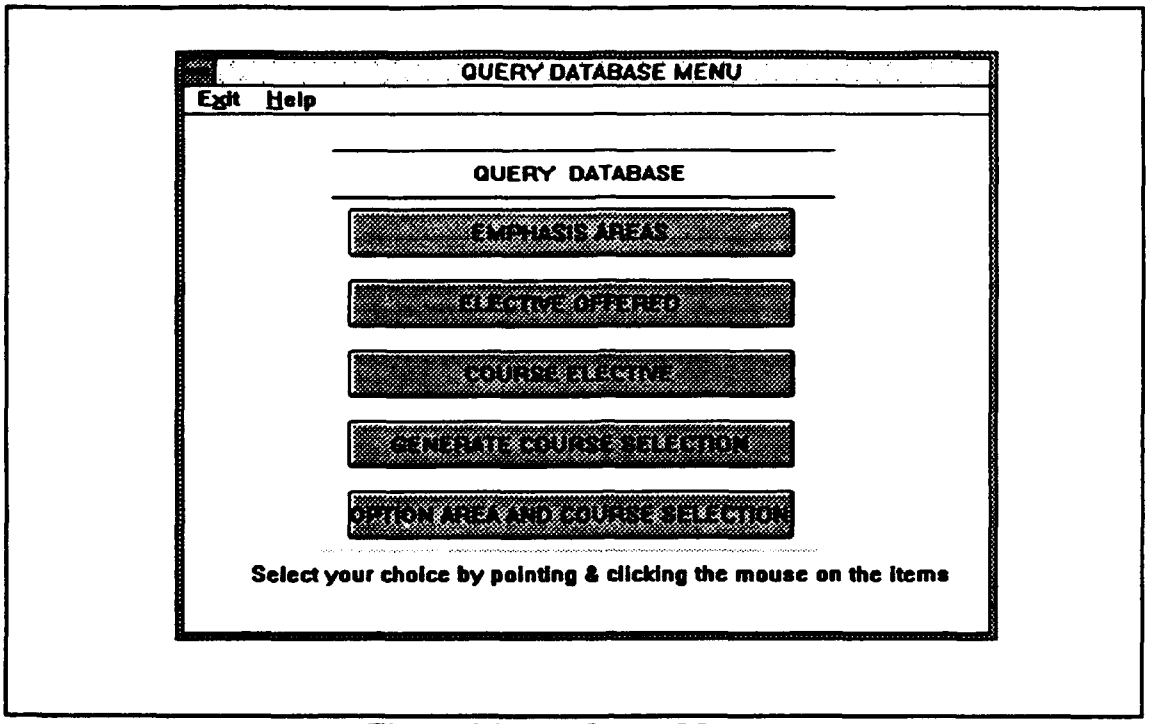

Figure 3.5 The Query Module

options. For instance, when we select the *Military_Data_Processing* option, the system will display a window frame shown in figure 3.6.

b. Query Elective Courses

The Query Elective module will produce the list of elective courses associated with given the emphasis area (see figure 3.7). Whenever the user want to know more information about specific course, then he or she can select the desired course from the lists. The system will give a course information shown in figure 3.8.

c. Query Elective Offered

The Elective Offered module gives a list of the elective courses offered in given specific quarter. For example, when we select the *Summer* option from the *Quarter_Name* pull-down menu and the *91* from the *Year* pull-down menu, the system

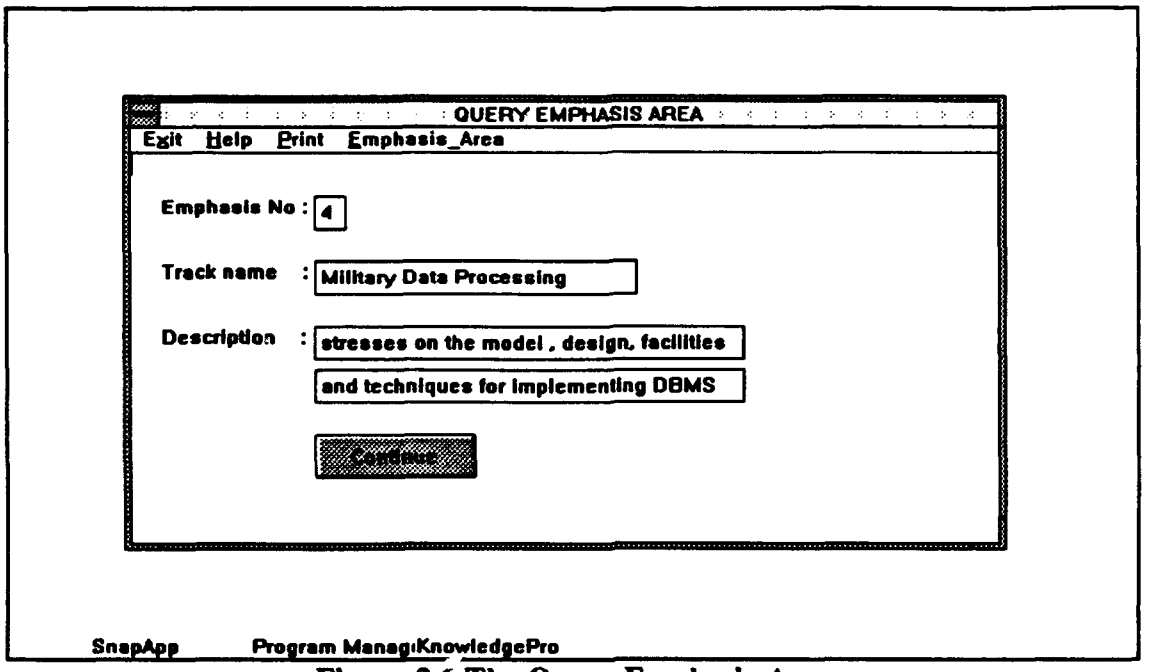

Figure 3.6 The Query Emphasis Area

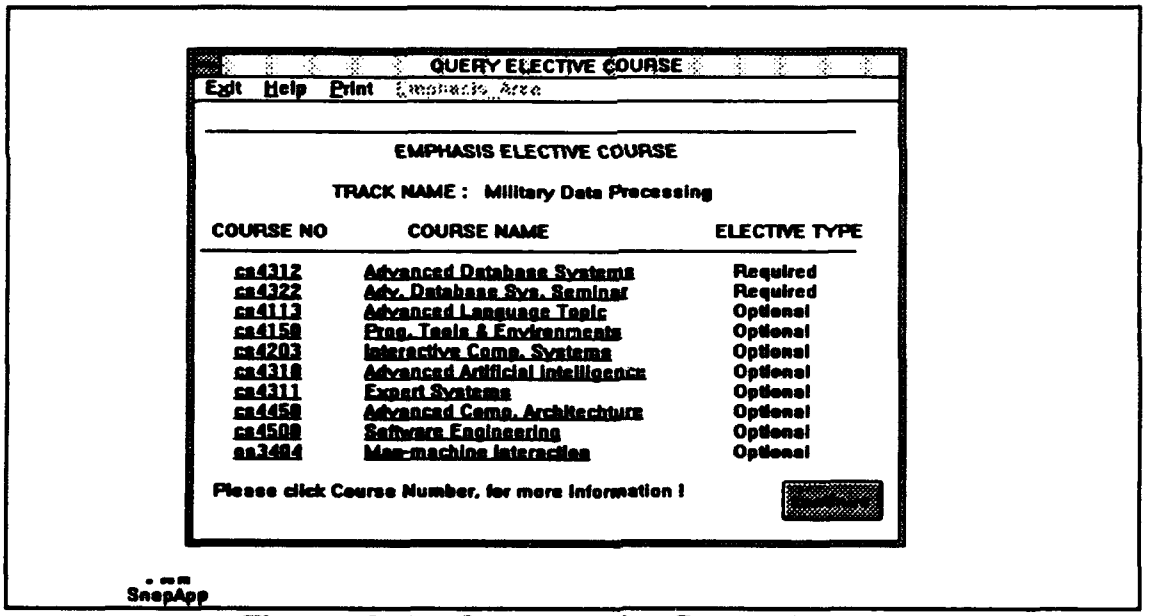

Figure 3.7 The Query Elective Course (frame 1)

will produce the course offering in summer 91 (see figure 3.9).

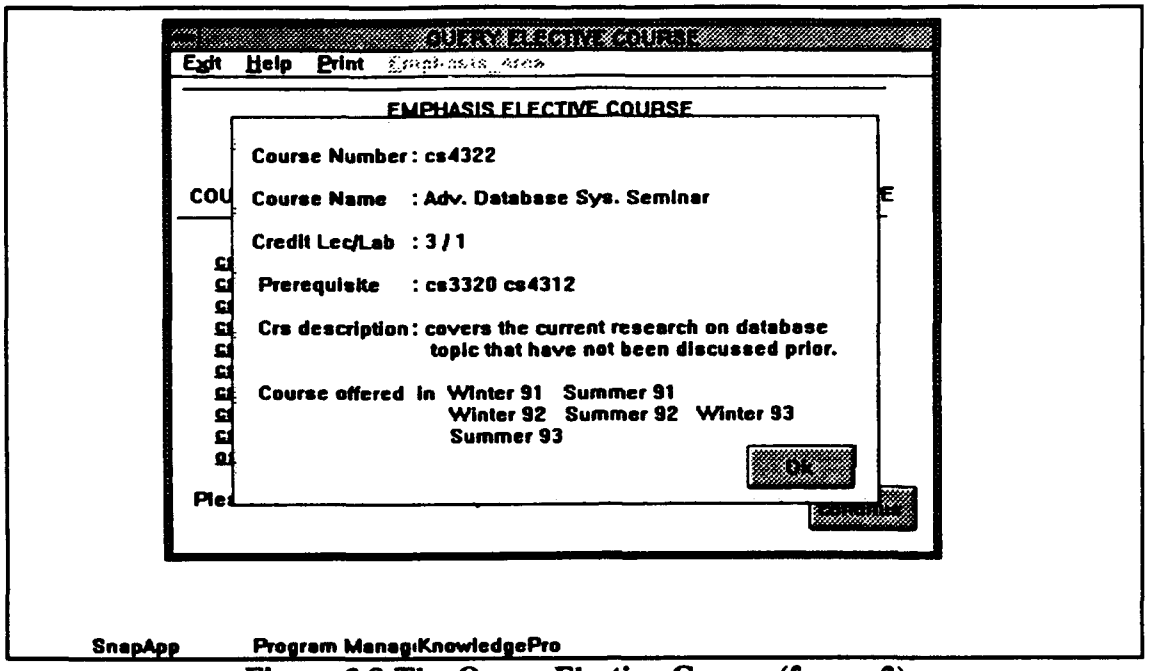

Figure 3.8 The Query Elective Course (frame 2)

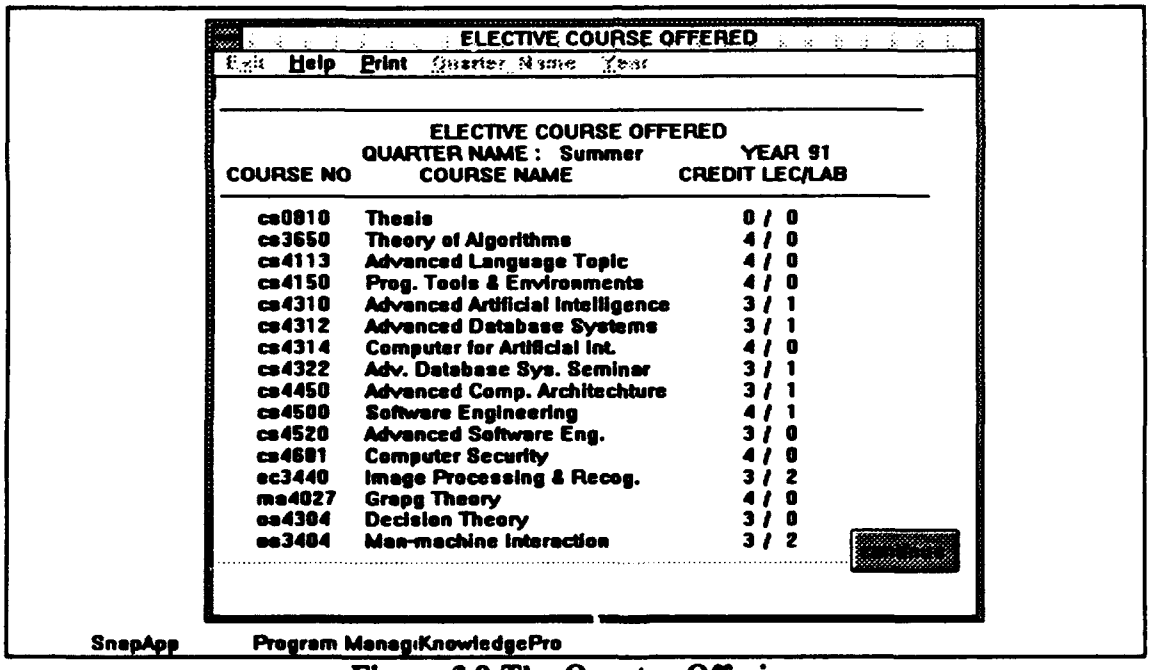

Figure 3.9 The Quarter Offering

d. Option Area and Course Selection

This module gives the final product of the system is report entitled "COMPUTER SCIENCE(368) EMPHASIS AREA DECLARATION AND COURSE SELECTION". It is the recommended courses suggested by the system. This report consists information about student's name and rank, class section, the emphasis area selected, the emphasis area electives requested and alternate elective requested for the final five quarters of the curriculum. The report is submitted by the student and is forwarded the Academic Associate and Curricular Officer and is entered in the Curricular Database to generate Schedule requests from Registrar's Office (see figure 3.10 and figure 3.11).

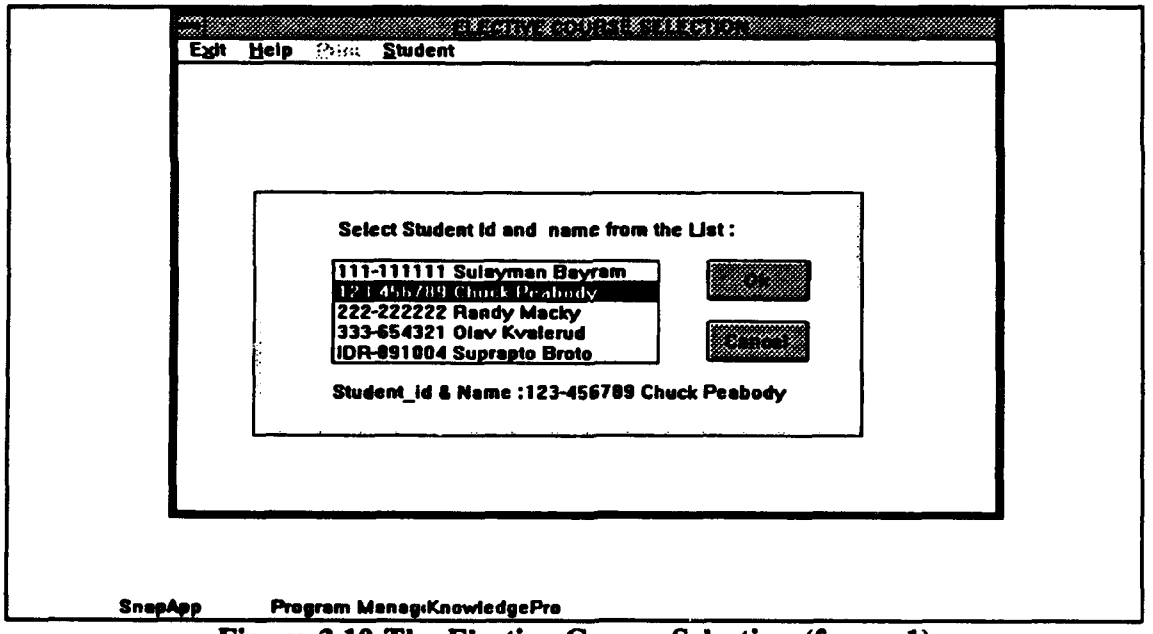

Figure 3.10 The Elective Course Selection (frame 1)

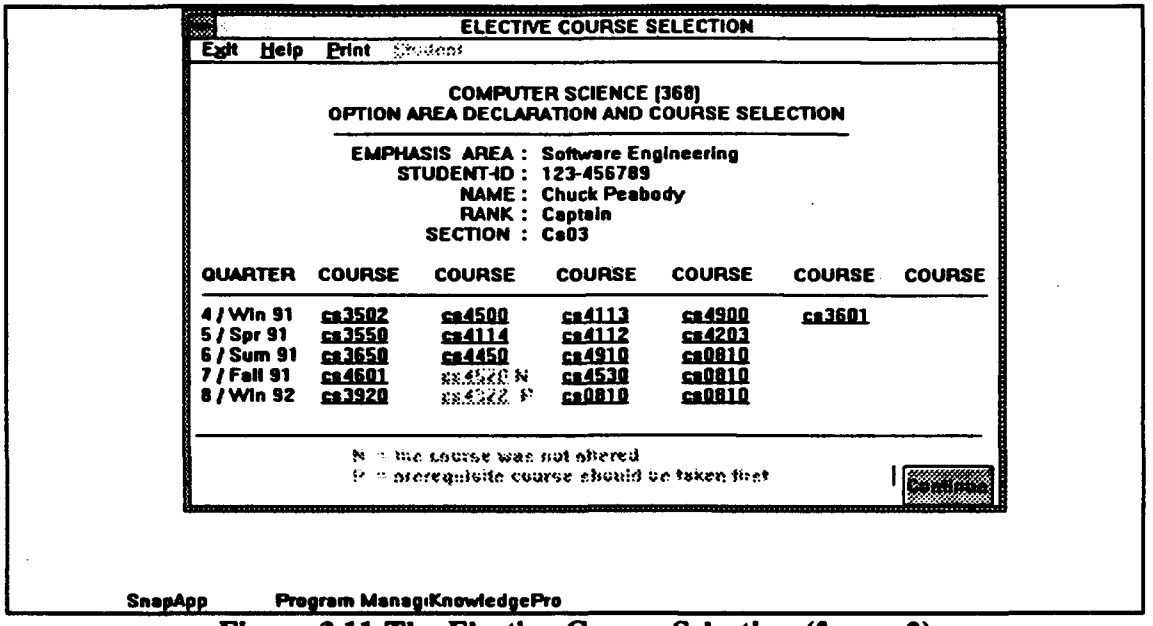

Figure 3.11 The Elective Course Selection (frame 2)

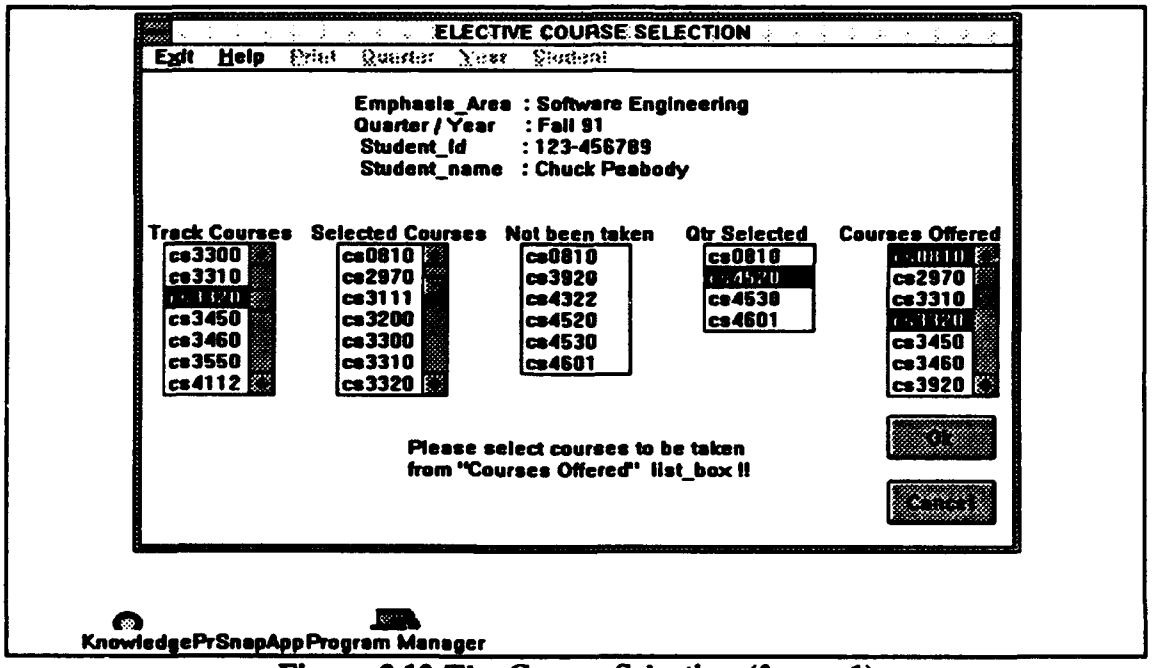

Figure 3.12 The Course Selection (frame 1)

e. Generate Course Selection

It is possible that the selected courses are not offered in the quarter the

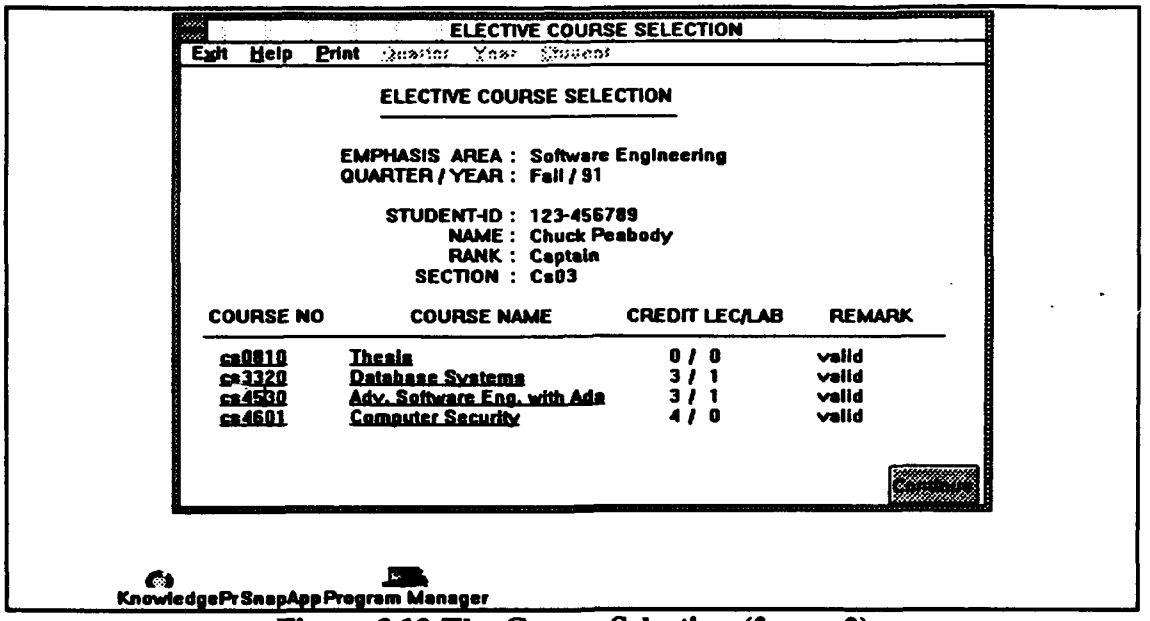

Figure 3.13 The Course Selection (frame 2)

user selects. In figure 3.11 the course number cs4520 will not be offered in Fall 91. The report also says that the student first should take the prerequisite course for cs4322. The Generate Elective Course selection module provides a modification mechanisms to change the undesired courses. There are several steps to update the undesired course selection shown in figure 3.11. First we select the *Fall* option from the *Quarter_Name* pull-down menu. We then select the student name from the student identification list. Finally we select the *91* option from the *Year* pull-down menu. The system will produce a window frame shown in figure 3.12. Instead of choosing cs4520, we replace it with cs3320. If we commit by clicking on the *Ok* button, the result will appear on the screen (see figure 3.13).

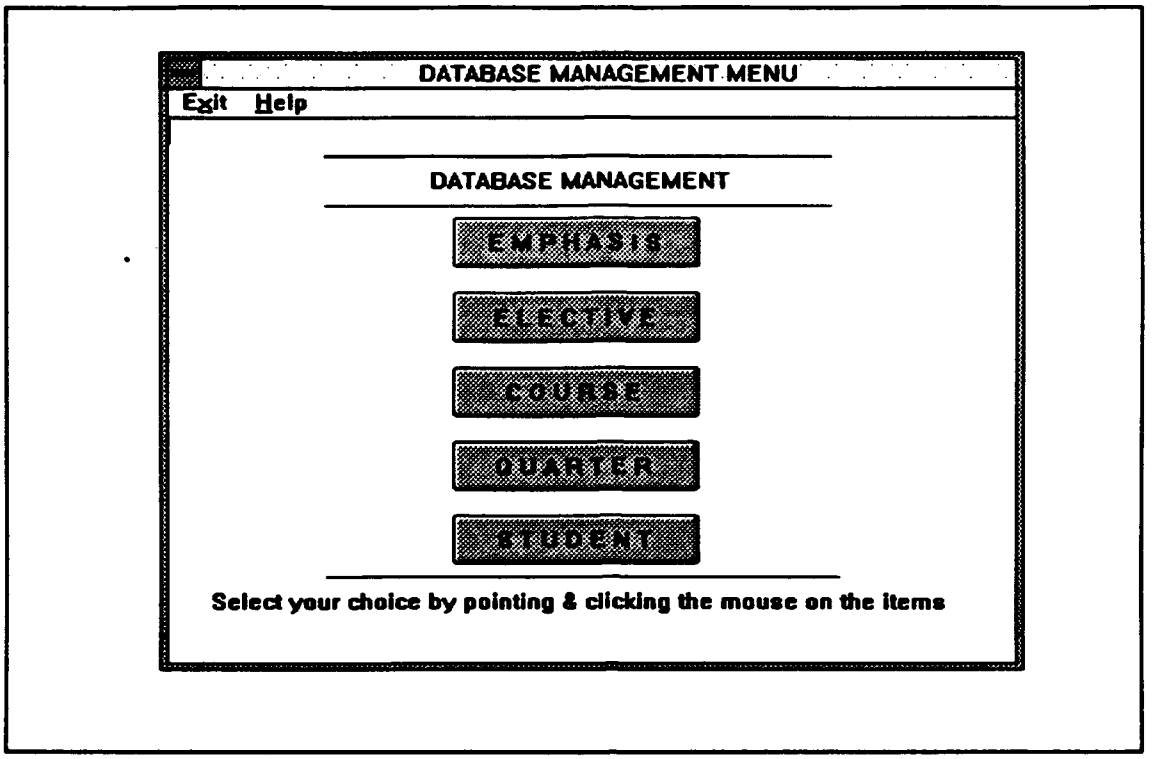

Figure 3.14 The Management Database Module

5. Database Management Module

As we stated before that the main menu consists of the *Query Database* and the *Management Database*. The *Query Database* has already been discussed, and now let we discuss the *Management Database*. There are five button options: *Course, Emphasis, Elective, Quarter* and *Student* options (see figure 3.14). By pointing and clicking the mouse to one of the buttons such as *Course*, the system will directly manipulate the *Course Object* as a result a sub menu for the *Course Object* will be displayed on the screen. Each sub menu of Management Database consists of three button options. These options are for adding new data, deleting data and updating the data associated with the file that stated in the button selection. By having the similar options in every sub menu

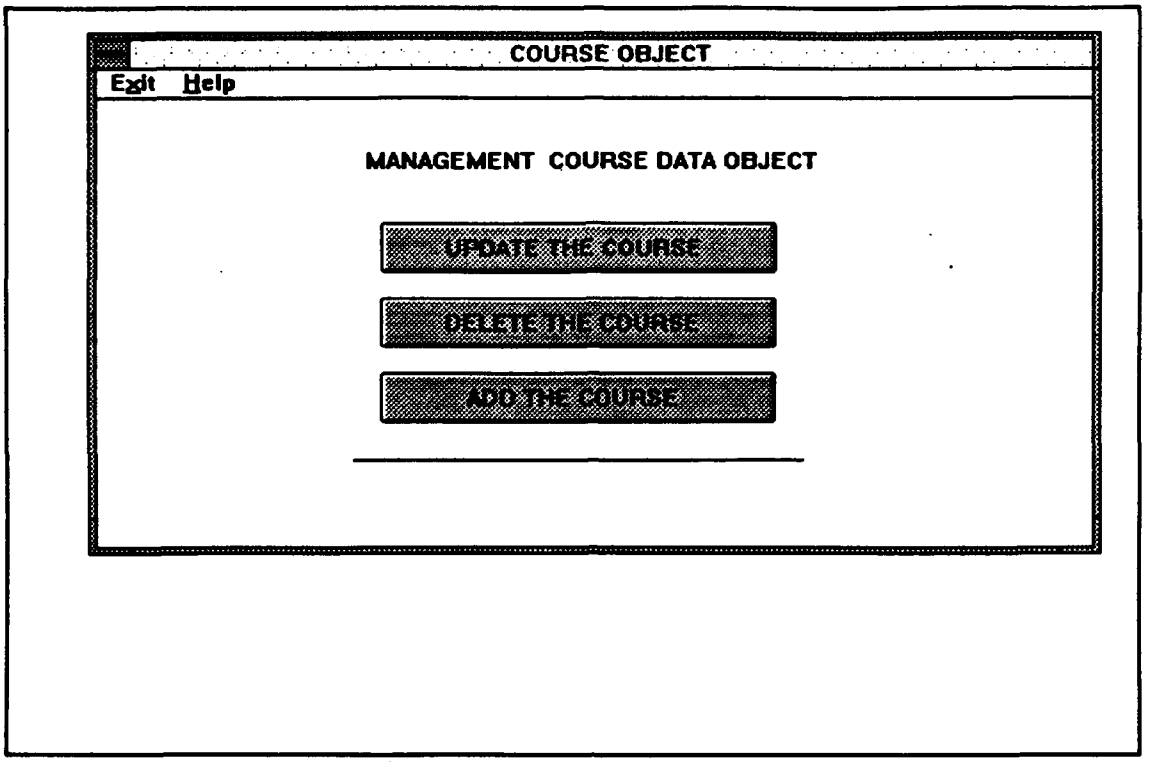

Figure 3.15 The Course Menu

and in the same order, this helps the user and reduces errors and training time.

In this section, for simplicity, we only discuss one example of Management Database, which is the *Course Object.* When we select the *Course* button of the Management Database menu, then it will display the sub menu of Management Course Data Object on the screen (see figure 3.15). This menu provides three button options : *Update the Course, Delete the Course* and *Add the Course.* These options are for updating mechanisms. The *Exit* option is for ending this session and the *Help* options is for getting information about the Course Object module. Now let us select *Update the Course* option to change the current record in the Database.

The window frame shown in figure 3.16, is the result given after clicking the

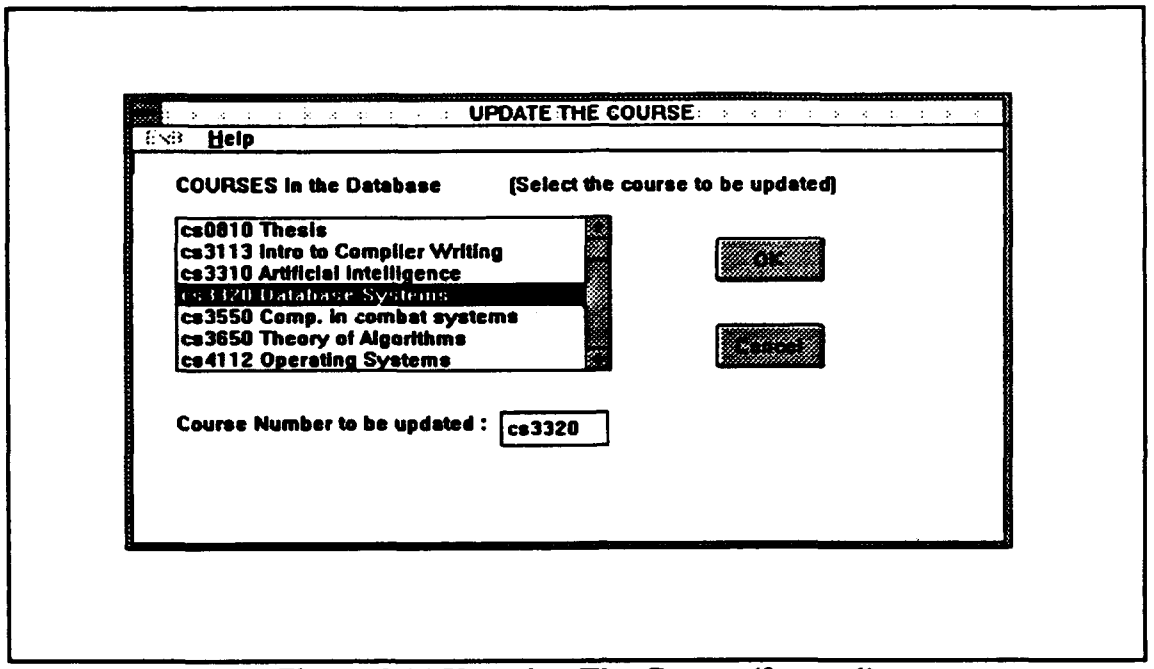

Figure 3.16 Updating The Course (frame 1)

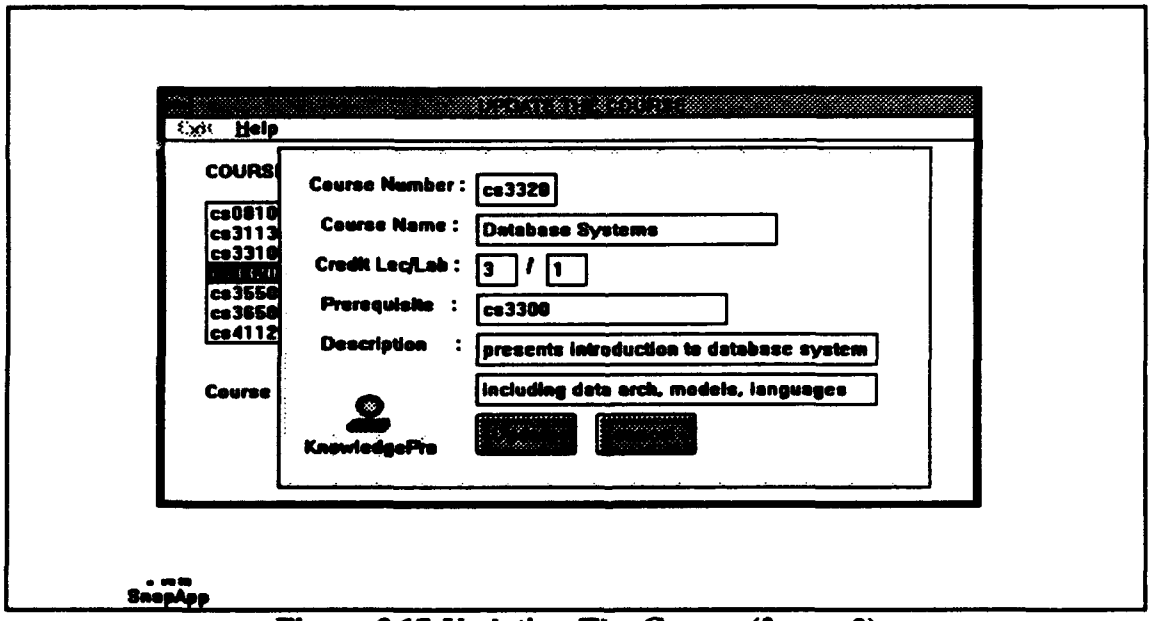

Figure 3.17 Updating The Course (frame 2)

Update the Course option. Now, the user can select the course to be updated from the list-box with the list of courses. Using a single click on the course followed by clicking

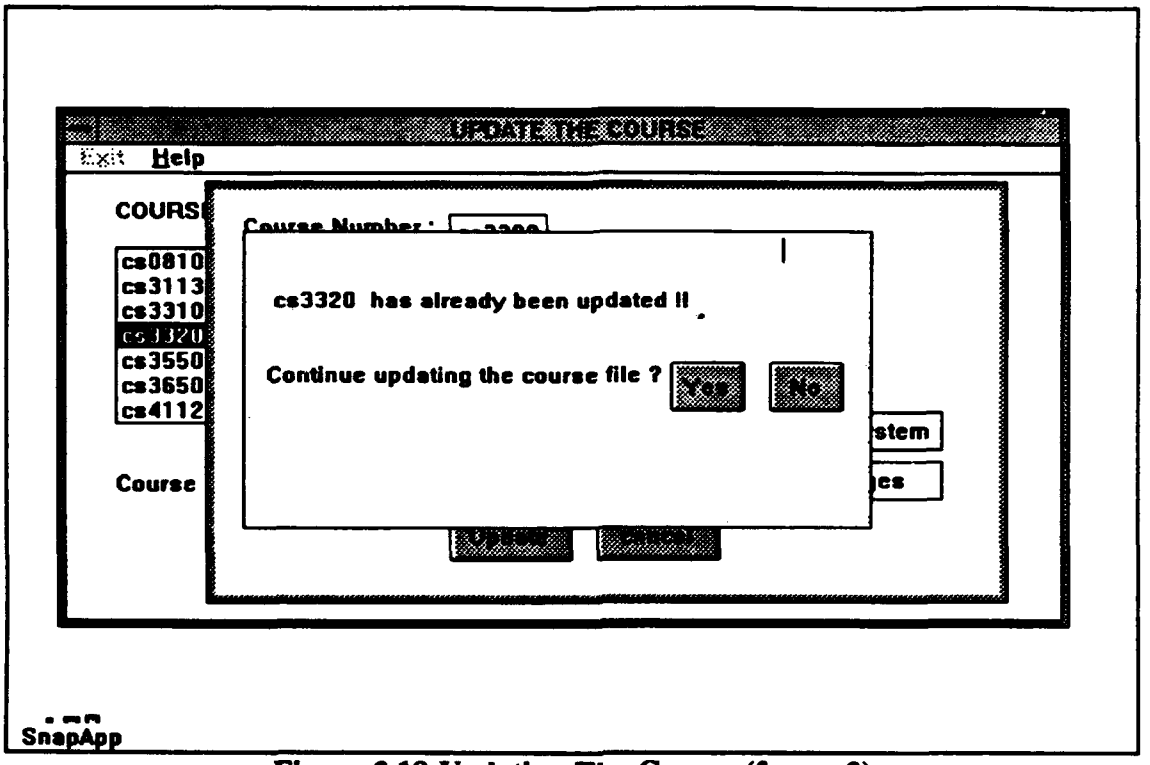

Figure 3.18 Updating The Course (frame 3)

on the *Ok* button or using the double clicking the user can update directly. The system immediately gives a response by displaying detailed information of the record selected. Figure 3.17 depicts overlap windows, where the user can edit and modify the data from current windows. This menu provides *Update* button and *Cancel* button which allows to the user to process or cancel the operation. After the user finish updating the selected course, a message window will be displayed. This message window gives the user options to continue updating Course files or not (see figure 3.18). Deletion and addition can be done in the same manner.

6. Coding the Programs

As we stated earlier, that a program created by the user in the KnowledgePro environment is called Knowledge base. Through the Knowledge base, the system communicate to the user and, can also provide a way for the user to respond. In other words, all functions of the EAIS are controlled by the Knowledge bases interactively.

There are a total of eleven Knowledge bases in the system. Each controls the execution of one or more modules. Table 3.1 depicts the Knowledge bases and functionalities used in the system.

Table 3.1 THE EAIS KNOWLEDGE BASES

Knowledge base	Purpose
MAIN.KB	- Display the Main Menu. - Display the Query Database menu. - Display the Management Database menu.
EMPHASIS.KB	- Display the menus for managing the Emphasis object. - Control the Add, Update, Delete, Help, Exit options. - Handle for managing EMPHASIS record.

ELECTIVE.KB	- Display the menus for managing the Elective object. - Control the Add, Update, Delete, Help, Exit options. - Control and manage the ELECTIVE record.
COURSE.KB	- Display the menus for managing the Course object. - Control the Add, Update, Delete, Exit options. - Control and manage the COURSE record.
QUARTER.KB	- Display the menus for managing the Quarter object. - Control the Add, Update, Help, Exit options. - Control and manage the QUARTER record.
STUDENT.KB	- Display the menus for managing the Student object. - Control the Add, Update, Delete, Help, Exit options. - Control and manage the STUDENT record.
QUEMPHAS.KB	- Display the menu for querying the Emphasis Area. - Control the Emphasis, Help, Print, Exit options. - Print the emphasis area report.

QUELEC.KB	- Display the menu for querying the Elective course for selected the Emphasis Area. - Control the Emphasis, Help, Print, Exit options. - Print the track elective courses report.
OPTADEC.KB	- Display the menu for querying selected courses from fourth through eight quarter. - System will view fourth through eighth quarter courses by given student's name selected from list box menu.
OFFERED.KB	- Display the menu for querying the Elective course for given Quarter. - Control the Quarter, Help, Print, Exit options. - Print the quarter elective courses report.
GENSELEC.KB	- Display the menu for selecting courses. - Given the quarter year, student's name, the system will view the list of courses should be selected. - Generate and print the course selections.

Through our experience of implementing a EAIS application using a software development tool KnowledgePro(Windows), we found some advantages as well as disadvantages. In the next chapter we will try to evaluate the development process of the EAIS using KnowledgePro.

IV. EVALUATION

In the previous three chapters, we have described some motivation and background of the thesis, and implementation of the EAIS. Now we will evaluate the application development process of the EAIS using toolkit KnowledgePro(Windows). We organize this chapter into three sections. Section A will discuss the category of KnowledgePro (Windows). Based on our experience, section B will discuss the advantages and disadvantages of using KnowledgePro. The last section will look at the comparison between KnowledgePro and Toolbook.

A. THE VISUAL INTERFACE CATEGORY OF KNOWLEDGEPRO

As we mentioned earlier, we know that visual programming growing in two major directions. In one direction, pointing devices and graphical techniques are used to obtain the visual environments for program development and execution; for software design; and for information retrieval. They all provide a visual environment which captures a new way for humans to interact with the computer. In the another direction, languages are designed to handle visual information; to support visual interaction; and to program with visual expression. Programming languages that fall into this category support a way in terms of an approach to the language aspects of the programming process. Table 4.1 shows the trends of the current visual programming.

We have briefly examined these different aspects of visual programming. The purpose of the classification is to sharpen our understanding of visual programming by

focusing on the functional distinctions; and to have the distinctions among the different categories of visual programming. To classify whether KnowledgePro is visual programming or not, is the consideration of graphical representations must be used in the programming process. Now let we discuss the aspects of KnowledgePro that can be applied by visual programming.

Table 4.1 VISUAL PROGRAMMING

VISUAL ENVIRONMENT				VISUAL LANGUAGES				
visualization of data	visualization of program / execution	visualization of soft. design	visual coaching	for handling visual information	for supporting visual interaction	Visual Programming Languages		
						Diagramatic systems	Iconic systems	Form system

Some capabilities supported by KnowledgePro include linking screen objects to topics, displaying a picture, handling events, getting information from the user, and temporarily halting topic execution.

1. Linking screen objects to topics

In the Main Menu (figure 3.4), the two options on the screen are located on buttons. There are the Query Database and the Management Database buttons. A button is used to triggered an action. When a button is selected something happens immediately. These are both examples of screen objects. The highlight is moved among screen objects

by pressing TAB or SHIFT TAB or by clicking on an object with the mouse.

The command that creates the button shows how an event topic is assigned to screen object.

button (QUERY DATABASE, QueryDatabase, 18, 10, 30).

The first parameter defines the name on the button, the second defines the event topic and the final parameters set the column, row and length where the button is displayed. So when an event topic is called it is passed these piece of information : information about the event; the name of the event; and the handle of the current screen object. Something called a *select_event* occurs as the default event for s button. When the *select_event* occurs, the button's event topic, *QueryDatabase*, is immediately called.

2. Displaying a picture

There two different types of graphics supported by KnowledgePro. Each of these is used for a different function. The first type is Icons. Icons are images associated with windows. The second one is bitmaps. Bitmaps are used for displaying pictures.

Bitmaps can be created using Zsoft's PC Paintbrush for Windows. To load a bitmap, we use **load_bitmap** command. The following is an example command to load a bitmap that returns a bitmap photo :

photo is load_bitmap ('PICTURE.BMP').

We then may display, print and delete using the commands :

bitmap (?photo).

print ('#g',?photo).

delete_bitmap (?photo).

In the same way, the icons can be created using software called ICOEDIT. For instance, to load the icon LOGO.ICO, display it at column 4 row 5 of the current window and delete an icon, we use the following commands:

```
logo is load_icon ('LOGO.ICO').
icon (?logo, 4, 5).
delete_icon (?logo).
```

3. Handling events

We have mentioned in chapter II, that KnowledgePro is an event driven language. When we view a program as a communication between the system and the user, this means that via the system the designer talks to the user and then waits for the user to talk back. When the user talks back, an event occurs. When we create a screen object, we specify whether we want it to handle events and which events we want it to recognize. This is done by specifying an event handling topic. Event handling topic is called whenever one of the specified events occurs while the focus is on the particular object.

To make it clearer, let's look at the following example.

```
w1 is window(WindowTopic,20,3,50,10,title,,,,[move_event,close_event]).
topic WindowTopic (info,event,handle).
    do (?event).
    topic move_event.
        column is first (?info).
        row is last (?info).
    end.
```

topic close_event.

WindowTopic is true.

end.

end.

When window w1 was created we specified that *move_event* and *close_event* should be recognized. When one of these events occurs, then *WindowTopic*, the event topic defined in the window command is called. When the user moves w1 a *move_event* occurs.

4. Getting information from the user

During visual user interface development, it is important that the selection of screen objects are the one best suited for the information being gathered. KnowledgePro provides edit objects that let the designer gather information such as name, class, rank quickly and easily.

The commands or functions to create edit objects include:

- *edit_line* is used for entering a single line of data
- *edit_box* is used for entering multiple lines in a dialog window
- *edit_window* opens independent editing window
- *edit_file* opens independent window with a File, Edit and Search menu

The user can select one or more items from a list of options using

- *list boxes, check boxes* or *radio buttons* that let the user selects one or more items
- *hyper-regions* used for selecting values using pictures

KnowledgePro provides many types of *screen objects*. It makes the designer

selects among them to make the best use of system environment. Additionally, each screen object has the same behavior make it very obvious to the user what has been selected and they provide visually attractive way to make options apparent.

5. Temporarily halting topic execution

There are times when we need to temporarily halt the execution of the commands. This can be done with *wait* command. Generally, the use of *wait* is to halt the system to wait for the user input as the following example:

```
window ( ).
text ('Welcome to the #mNaval Postgraduate School#m').
wait ( continue, 5 ).
```

A window is opened and a welcoming message is displayed. *wait* then stops executing the system until five seconds have expired or until the user select the *continue* button in the wait window, whichever come first. When the user select the hypertext, topic *Naval Postgraduate School* is called.

KnowledgePro may be classified as a language that can be used to create, define, and manipulate graphical representation. The objectives of developing KnowledgePro are motivated by the need to have easy-to-use languages for the manipulation and querying of graphical data. KnowledgePro provides a way to deal with all possible sequences by the user with the input devices. This means that KnowledgePro is a user interface development tool that exploits the concurrency among multiple interaction devices such as mice, buttons, list boxes, radio buttons, and keyboards. KnowledgePro provides the

capability to create every kind of control available in the windows environment, it makes for graceful interaction between the user and the computer.

Based on the capabilities of KnowledgePro, we believe that KnowledgePro focus is on language aspects for a good visual user interface. As mentioned before, visual languages can be classified into language for handling visual information, language for supporting visual interaction, or language for programming with visual expression. KnowledgePro falls into the second category, Languages for Supporting Visual Interaction.

B. THE ADVANTAGES AND DISADVANTAGES OF USING KNOWLEDGEPRO

The goal of this thesis is to study the effectiveness of a visual interface. To accomplish the goal we have implemented the EAIS using toolkit KnowledgePro. Through our experience we have found some advantages and disadvantages of using KnowledgePro.

1. Advantages

As we mentioned before that the decreasing cost of computing, coupled with the widespread use of personal computers, has acted as a catalyst for more application. By necessity, end user-computing is becoming a major trend.

KnowledgePro offers characteristics to enhance the end-user computing, reach a broader personnel base of programmer; and increase their productivity. Some of these characteristics are: ease of learning; ease of use; and modularity.

a. Ease of Learning

KnowledgePro comes with a single, large manual that includes tutorial, reference, and appendix sections. It uses a multiple document interface and also provides a large application hypertex-based on line help that let users work on more than one application at a time. If command's errors are found, we can directly call on line help without quitting from current application. KnowledgePro also includes the complete files for KnowledgePro tools, such as FONT.CKB and DESIGN.CKB which serve as excellent references for experience programmers. From our experience of using KnowledgePro in the EAIS implementation, we easily learn and implement it. Reducing learning time can increase productivity and number of people who can program.

b. Ease of Use

Traditionally, computer programming languages that were developed in the 1960s, 1970s and early 1980s, such as FORTRAN, COBOL, ALGOL, PASCAL, ADA, or C were designed for use in the non-interactive computer environment. Programmers would compose hundreds or thousands of lines of code, carefully check them over, and then compile or interpret the code by the computer to produce a desired result. Creating a visual interface in these traditional approaches requires unacceptable amount of time to design, code, test and maintain the software.

KnowledgePro is a mainstream object-oriented Windows development system that offers additional hypertext and expert system tool feature to enhance the visual user interface. KnowledgePro works as integrated development under Windows. The KnowledgePro main window includes commands for maintaining, compiling and

running files; searching and replacing text in edit window; running the application; clearing the current application; debugging the application in memory; and a switch for setting a slew of KnowledgePro environment options. The main window also offers an on-line context-sensitive hypertext help function.

The KnowLedgePro language is surprisingly compact for such a powerful system. It contains few syntax rules, so we can be verbose in our coding, without destroying the meaning of the topic or encountering an error. For manipulating data, KnowledgePro provides string and financial operations as well as facilities for performing set operations against lists. KnowledgePro includes the source code for the help system application. Creating and manipulation push buttons; radio buttons; check, list and edit boxes; and hypertext region is easy. It also provides several tools and their source code, to make it easier for users to design and build an application. FONT.CKB and DESIGN.CKB are the most important function as interactive code generation tools to emphasize speed on implementation.

The debugger includes several tools, including a window for displaying application topics. We can use it to browse a topic's values, commands and properties, much like an object browser in Smalltalk. It also provides a command to evaluate the text of the selected edit window; a command to compile and execute the code immediately and a command to trace each step as the application is executed. The *Calls* window displays a list of topics that our application has called but not yet executed.

KnowledgePro offers solid and useable development tools that run very quickly and cleanly. This is one Windows development environment tool that offers the

ease of use, the ease of learning and extensibility for both power users and developers.

c. Modularity

The principles of software engineering recommend writing code in small, self-contained units, called modules. Modularization is the process of dividing a task into subtasks. Each module performs a separate, independent part of task. Modularity offers advantages for program development, as well as software maintenance.

A knowledge base is a collection of topics. Topic strongly supports modularity. When KnowledgePro is loaded, only one topic, named *!main*, exists. If we compile a knowledge base, the commands in the knowledge base become associated with *!main*; topics defined in the knowledge base become subtopics of *!main*. Topics can also be subtopics of other topics. This is called nesting. There are several advantages to writing a program as a series of topics.

- Understandability. The topic structure groups together related functions and hides the details of their implementation. This makes debugging easier and the knowledge base more understandable.

- Maintainability. If a function is implemented as a single topic, the topic can be replaced with a revised one, if necessary. The new topic may be needed due to a change in requirement or the environment.

- Reusability. Topics developed for one purpose can often be reused in other programs. Reuse of correct, existing topics can significantly reduce the difficulty of programming and testing.

2. Disadvantages

a. Speed

KnowledgePro offers the ease of use by providing many kinds of different screen objects such as push buttons, radio buttons, check boxes, edit boxes, list boxes, edit line and hypertext regions. The designer can build user interfaces easily. However, creating and manipulating the screen objects, make the system runs slowly. From our experience, the primary difficulty of implementing the EAIS using KnowledgePro is achieving quick navigation through the system. This lack of speed is due to the large overhead involved in processing screen objects by the Central Processing Unit.

The solution of this problem is to invest in hardware that speeds up graphics processing or provide the system with other toolkits such as KnowledgePro Math Toolkit, or KnowledgePro Graphics Toolkits.

b. Error Handling and Compiling

When we compile a knowledge base with many errors, KnowledgePro will display an error at a time on window screen. The system wait for the response of the user to continue the operation or cancel. By displaying one error at a time on the screen, the system needs more time and is not attractive to the novice programmers and end-users. Lets us consider the following knowledge base :

```
topic first.

    list is [A, B, C].
    item1 is first ( ?list ).

end.
```

When we compile it, we do not have any errors. But when we run this program, the result is an infinite loop because it has the same name as a function which is used in a command for the topic. Instead of calling the function *first*, the knowledge base *topic first* is recursively called forever.

c. Design Tools lacking

Despite the powerful language and functions, we found KnowledgePro's application design tools lacking. The tools include a simple font generator and an interface design tool. The interface design lets us the user create, position, and resize Windows objects in a window, generate the KnowledgePro code to create and manipulate them, and copy the code via the clipboard, to the user application. The font generator lets the user configure custom fonts from the available windows fonts.

The tools are rather limited and operate in nonstandard way. For example, most Graphical User Interface design tools lets the user use the mouse to put the object into position in the interface. With the KnowledgePro's interface design tool, when we click where we want the object to be, then the object jumps to where we clicked. Further, we have to use keyboard to resize an object because the mouse does not support resizing.

C. COMPARATIVE STUDY BETWEEN TOOLBOOK AND KNOWLEDGEPRO

As we stated before, that Toolbook and KnowledgePro are software development tools for Windows 3.0. The implementation of a sample application using KnowledgePro has already been discussed along with it's advantages and disadvantages. In this section,

we try to compare both tools to have a better understanding for software development tools.

1. ToolBook

Toolbook greets the user with its Book-shelf front end. A "book of books" showing icons for the available Toolbook applications. The user is off to a quick start with the following applications: the DayBook information manager; the ToolBook/DBASE Exchange; a graphics front end to DBASE files; and the Quick Tour. Both the ToolBook Quick Tour and the on-line help give the user easy access to any part of the system. This means that ToolBook offers access to the graphic power and intertasking linkage of the Windows environment. Support for basic Windows features of ToolBook and the ease of learning to use development tools are better in Toolbook than in KnowledgePro. ToolBook is an active environment, sending a constant stream of messages to its objects. This allows the designer to easily alert the user to problem areas on screen. A feature of growing importance in fully graphic applications designer alerts. ToolBook's limit on "nested handlers" forces a recursive benchmark to be written in an interactive style. In contrast, KnowledgePro accepted recursive benchmarks without rewriting them.

2. KnowledgePro

KnowledgePro works with "knowledge bases" in contrast with ToolBook's pages of books. KnowledgePro supports hypertext feature that can be clicked upon by a mouse to produce some programmed action. By making on screen text an active component of the interface, designers can build a single software object that serves as

both code and data. The result is extraordinary development productivity. Like ToolBook, KnowledgePro is an active environment in which objects receive a stream of messages as the user takes various actions. KnowledgePro is also similar to ToolBook in the way that individual objects are given handlers for those events that the programmer considers relevant to each object.

KnowledgePro is more responsive than ToolBook, with interactive touches such as editor windows that scroll interactively as the user drags the elevator box at the right side of the window. KnowledgePro offers full topic subclassing with multiple inheritance, a feature that distinguishes it from ToolBook. It's support for Windows objects is much richer than ToolBook's. KnowledgePro enables developers to produce Windows programs that offer broader functionality than those created in ToolBook.

V. CONCLUSIONS

Traditionally, the design and implementation of conventional database systems involve text oriented data access with their inherent lack of modularity, extensibility and modifiability. An alternative to this traditional approach is using visual interfaces for the design and implementation of databases. This alternative approach involves using software development tools (toolkits) to ensure modularity, extensibility and modifiability. To study the effectiveness of using visual interfaces, we have designed and implemented a sample application named The Emphasis Area Information System (EAIS) using KnowledgePro, a tool for rapid application under Windows.

An interactive user interface was implemented for the EAIS using KnowledgePro. An evaluation of our implementation indicates that designing an interactive user interface using visual interfaces is easier than designing a user interface using the traditional approach. KnowledgePro falls into the language for supporting visual interaction. KnowledgePro supports interactive design tools and easy-to-learn commands which extremely facilitates using and learning the system. Underlying this easy-to-learn environment is a powerful symbolic language called the topic. Topics are very highly support modularity, extensibility and modifiability which increase on speedy the system development.

The primary difficulty of the implementation of the EAIS using KnowledgePro is achieving quick navigation through the system. This lack of speed is due to the large

overhead involved in processing screen objects by the Central Processing Unit. The solution of this problem is to invest in hardware that expedites graphics processing or provide the system with other toolkits such as KnowledgePro Math Toolkit, or KnowledgePro Graphics Toolkits.

There are still some issues that need to be considered for future research. Some of these issues are the utilizing new features to Windows including:

- The expert system tool features to enhance the friendly user interface. The expert system capabilities enable the application designer to built in expert system rules that allow to be tailored to a user's level of expertise.
- Dynamic Data Exchange (DDE) as a method for applications that run under the Microsoft Windows to communicate directly with one another.
- Dynamic Link Library (DLL) is a separate file containing functions which can be called by the programs to perform specific jobs.

Using these features would increase efficiency in designing and implementing visual interface to database systems while decreasing the time.

APPENDIX A. SOURCE CODE LISTING

```
(*      Program        : main.kb
        Programmer : suprapto
*)

if exists(temp)
   then remove_topic(temp) and collect() and workon()
else preparation().

 topic preparation.
  logo().
 setup().
 Main_menu ().
end.

topic logo.
 wh is window(,15,4,66,19,,[Dialogwindow,visible]).
  text ('#n#fblue




                                COMPUTER SCIENCE

                            NAVAL POSTGRADUATE SCHOOL

                                MONTEREY CALIFORNIA 93943#d').
  b2 is button(OK,  ,28,16,10).
   set_focus(?b2).
     wait( ,1).
  close_window().
end.
(* <<<<<<<<<<<<<<<<<<<<<<<< MENU ITEMS >>>>>>>>>>>>>>>>>>>>>>>>>>>> *)

topic main_menu.
   disable_menu_item (?m1,[previous_menu]).
   list is [].
   list gets '&Help'.
   set_title ('MAIN MENU').
   text ('#e #fblue
            ---------------------------------------------------------------------- #fred

                            NAVAL POSTGRADUATE SCHOOL

                      EMPHASIS AREA DECISION SUPPORT SYSTEM#fblue

            ---------------------------------------------------------------------- ').
     set_display_pos(9,16).
```

```
    text('#fblue----------------------------------------------------------------------------------#d
        Point and click your mouse to select job on the items above:

').
button('QUERY DATABASE','query database',18,10,30).
button('DATABASE MANAGEMENT','database management',18,13,30).
end.

topic workon.
  data is read('tempo.dat'). setup().
  if first(?data) is 1
     then do('query database')
   else do('database management').
end.

topic select (item).
  do (?item).


   topic 'Quit_to_system'.
     text(#e).
      logo().
      clear().
   end.   (* &Quit *)
   topic &Quit.
     text(#e).
      logo().
      clear().
   end.   (* &Quit *)

   topic &Help.

      items is last(?list).
      display_Info(?items).

    end. (*&help*)

end. (*SELECT*)

(* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> *)

topic 'QUERY DATABASE'.
   list gets 'QUERY DATABASE'.
   set_title (?w1,'QUERY DATABASE MENU').
   enable_menu_item (?m1,[Previous_menu]).
   text ('#e #fred
                  ----------------------------------------------------------------------#fblue
                                    QUERY  DATABASE  #FRED
                  ---------------------------------------------------------------------- #D').
   set_display_pos(9,19).
    text('#FRED    ----------------------------------------------------------------------#D
        Select your choice by pointing & clicking the mouse on the items
').
```

```
  button('EMPHASIS AREAS',emphas,15,5,38).
  button('ELECTIVE OFFERED',offered,15,8,38).
  button('COURSE ELECTIVE',elective,15,11,38).
  button('GENERATE COURSE SELECTION','generate',15,14,38).
  button('OPTION AREA AND COURSE SELECTION',optadec,15,17,38).
end. (*QUERY DATABASE *)

topic emphas. mark('emphasis areas'). end.
topic offered.  mark('elective offered'). end.
topic elective. mark('course elective'). end.
topic generate. mark('generate course selection'). end.
topic optadec. mark('option area'). end.

topic 'DATABASE MANAGEMENT'.
   list gets 'DATABASE MANAGEMENT'.
   set_title (?w1,'DATABASE MANAGEMENT MENU').
   enable_menu_item (?m1,[Previous_menu]).
   text ('#e #fred
                  ----------------------------------------------------------------------#fblue
                                DATABASE MANAGEMENT #fred
                  ----------------------------------------------------------------------#d').
        set_display_pos(12,19).
text('#fred  ----------------------------------------------------------------------#D
     Select your choice by pointing & clicking the mouse on the items ').
      button('E M P H A S I S','dm_emphasis',24,5,20).
      button('E L E C T I V E','dm_elective',24,8,20).
      button('C O U R S E','dm_course',24,11,20).
      button('Q U A R T E R','dm_quarter',24,14,20).
      button('S T U D E N T','dm_student',24,17,20).
end. (*QUERY DATABASE *)

topic dm_emphasis. mark('e m p h a s i s'). end.
topic dm_elective. mark('e l e c t i v e'). end.
topic dm_course. mark('c o u r s e'). end.
topic dm_quarter. mark('q u a r t e r'). end.
topic dm_student. mark('s t u d e n t'). end.

(* ========================= MARK TOPIC ========================= *)

topic  mark (item).
  text(' tes2'). text(?item).
  if one_of(?list_menu, ?item)
    then application()
   else DO (?ITEM).

 topic application.
      hide_window(?w1).
      load ( element ( ?list_prog, where ( ?list_menu, ?item)), temp).
      temp().
      wait().
  end.
 END. (*end mark*)
```

```
topic Previous_menu. main_menu(). end.

topic display_Info (items).
   wh is window(,15,5,66,19,,[Dialogwindow,visible]).
   use_font(system_font).
    make_modal(?wh).
   text (#e,read ('info.hyp', concat ('//',?items) ,'//') ).
    set_file_pos ('info.hyp',0,beginning).
    b2 is button(Ok,continue,30,16).
    set_focus(?b2).
     wait().

     close_window().

  end.


(* ========================================================== *)


(* -^-^-^-^-^-^-^- SETUP ^-^-^-^-^-^-^-^-^-^-^-^-^-^-^-^-^-^-^-^-^-^-^-^ *)

topic setup.
  displayInfo ().
  if version is 3
     then !main:w1 is window (select:&Quit,13,1,83,22,,[Popup,Visible,ThinFrame])
                  e l s e      ! m a i n : w 1      i s      w i n d o w
(select:&Quit,13,1,70,25,,[PopUp,,ControlMenu,thickFrame,ShowChildren,Siblings],,,,close_event).
  m1 is menu ([[E&xit,Previous_menu,'Quit_to_system'],&Help],select).
  disable_menu_item (?m1,[Previous_menu]).
  list_menu is ['emphasis areas','elective offered','course elective','generate course selection','option area',
                  'e m p h a s i s','e l e c t i v e','c o u r s e','q u a r t e r','s t u d e n t'].
  list_prog is ['quemphas.ckb','offered.ckb','elecqry.ckb','genselec.ckb','optadec.ckb','emphasis.ckb',
                 'elective.ckb','course.ckb','quarter.ckb','student.ckb'].
  show_window (?w1).
  list is [ ].


  topic displayInfo.
    system is system_info ().
    if element (?system,4) is 12
      then displayType is EGA and
            smallFont is  [10,5,400,f,f,f,0,1,34,Helv] and
            bigFont is [16,9,400,f,f,f,0,1,18,'Tms Rmn']
      else displayType is VGA and
            smallFont is [16,7,400,f,f,f,0,1,18,Helv] and
            bigFont is [20,8,700,f,f,f,0,1,34,Helv].

    if last (?system) < 3
      then hyperColor is black
      else hyperColor is blue and
            hyper_display (blue).
    hyperFont is create_font ([12,8,400,f,t,f,0,1,2,System]).
```

```
   version is string_copy (element (?system,9),1,1).
 end.

end. (*setup*)

(* END MAIN.KB    *)

( *-----------------------------------------*)


(*       Title    : emphasis.kb
         Author : suprapto

*)

wmenu is window(select:&quit,5,5,82,21,,[Popup,,ControlMenu,ThickFrame,ShowChildren,Siblings]).
m1  is menu([[E&xit, 'Previous_menu', 'Quit_to_system'],&Help],select).
set_title(?wmenu,'emphasis OBJECT'). written is 0.
set_title(?wmenu,'EMPHASIS AREA OBJECT'). test is 0.
read_file().
topic read_file.
    eof is number_to_char(26).
     No is 0. emphasisField is []. emphasisList is [].
     message is read_line('track.txt',4).
     repeat
        emphasisField gets ([?message]) and
         emphasisList gets(element(?message,2)) and  No is ?No +1 and
        message is read_line('track.txt',4)
     until ?message is ?eof.  close_all().  show_window(?wmenu).

end.

selection().

topic selection.
       text(#e). enable_menu_item(?m1,[E&xit]).
       text ('#n#fred

                                  MANAGEMENT   EMPHASIS AREA#FBLUE
                                  ----------#d').
       set_display_pos(16,15).
                text('#fblue              ----------').
       button('UPDATE THE EMPHASIS AREA ',&update,24,6,32).
       button('DELETE THE EMPHASIS AREA ',&delete,24,9,32).
       button('ADD THE EMPHASIS AREA ',&add,24,12,32).
       wait().
end.
(*
topic E&xit.
       test is 0.
      selection().
end.
*)
```

```
topic  select(item).

   do (?item).
end.

topic &update.
   set_title(?wmenu,'UPDATE THE EMPHASIS AREA').
   disable_menu_item(?m1,[E&xit]).  option is '#FRED updating'.
   displaybox().

topic displaybox.
      text(#e). set_display_pos(5,13). text('#fblue emphasis area to be updated :').
   set_display_pos(5,2).  ed1 is [].
   text('#fblue emphasis areas in the Database #fred          (Select the emphasis to be updated)').
    Lb is list_box([?emphasisList],input_data,5,4,40,,,t,[list_select_event,double_click_event]).
    ed1 is edit_line(,,35,13,30).
    button (OK,Ok,55,5,10). button(Cancel,can,55,9,10).
  end.
end.

topic input_data(item,info_name).
   ed1  is  ?item.  edfield is ?ed1.
    ed1 is edit_line(?ed1,,35,13,30).
  if ?info_name  is  'double_click_event'  then ed1 is get_text(?ed1) and
        editone is ?ed1 and  check_data().
end.

topic Ok.
   if not(get_text(?ed1) is []) then ed1 is get_text(?ed1) and
         editone is ?ed1 and check_data()
     else  input_data(get_list_box(?Lb)).

end.

  topic check_data().
    if ?option is '#fred Updating' then edit_emphasis()
      else ( if ?option is '#fred deleting ' then delete_emphasis()
               else add_emphasis()).
  end.

  topic edit_emphasis.
     if  one_of(?emphasisList, ?ed1)
       then edit()
     else
          message is concat(?ed1, '#fblue  was not  in the emphasis file !') and
          display_info().
  end.

  topic delete_emphasis.
     if  one_of(?emphasisList, ?ed1)
       then delete()
     else
           message is concat(?ed1,'#fred  was not  in the emphasis file !!') and
```

```
            display_info().
  end.

topic display_info.
   disable_window(?wmenu).
   disable_window(?wh).
   wh1 is window(,20,10,51,12,,[Dialogwindow,visible]).
   text('#n #n   '). text(?message).
   text('#n #n #n  Continue '). text(?option). text(' the emphasis file ?').
   button(Yes,yes,43,5,6).
   button(No,main_menu,43,8,6).
end.

topic yes.
     close_window(?wh). remove_topic(wh).   enable_window(?wmenu).
      close_window(?wh1). remove_topic(wh1). ed1 is ?editone.
    if ?option is '#fred deleting '  then  ( if not(?message is ' ') then &delete()
                                              . else ed1 is edit_line(?ed1,,35,13,10))
       else ( if ?option is '#fred adding'  then  &add()
                 else  ed1 is edit_line(?ed1,,35,13,10)).
end.

topic main_menu.
      close_window(?wh). remove_topic(wh).
      close_window(?wh1). remove_topic(wh1).
      enable_window(?wmenu).
     selection().
end.

topic get_data.
    ed  is element(?emphasisField, where(?emphasisList,?ed1)).
    ed2 is element(?ed,1).  ed3 is element(?ed, 3).
    ed4 is element(?ed, 4).
end.

topic edit.
    keyvalue is  ?ed1.
    get_data().
     set_title(?wmenu,'UPDATE THE EMPHASIS').
     FormatDisplay().
     button (Update,yesUpdate,20,14,10).
     button (cancel, notUpdate, 32,14,10).

  topic notUpdate.
       message is ' '. disable_window(?wh).  display_info().
  end.

 topic yesUpdate.
   ed1 is get_text(?ed1).
   ed2 is get_text(?ed2).
   ed3 is get_text(?ed3).
   ed4 is get_text(?ed4).
   set_display_pos(20,16).
```

```
    disable_window(?wh).
    if one_of(?emphasisList, ?ed1)
      then (if ?ed1 is ?keyvalue
                then updatedata()
              else message is concat(?ed1,'#fred   was in the emphasis file !#d')
                   and display_info())
    else message is concat(?ed1,'#Fred   was not in the emphasis file !  Please, add first') and
         display_info().
  end.
end.

topic updatedata.
     edrec is  combine(?ed1,?ed2,?ed3,?ed4).
     emphasisField is replace(?emphasisField, [?ed],[?edrec]).
    emphasisList is replace(?emphasisList,?edfield,?ed).
     written is 5.
     message is concat(?ed1,'#fblue   has already been updated !!#d').
    disable_window(?wh).
     display_info().
end.

topic &delete.
    disable_menu_item(?m1,[E&xit]).
    set_title(?wmenu,'DELETE THE EMPHASIS AREA').
    displaybox().

topic displaybox.
       text(#e). set_display_pos(5,13). text('#fblue emphasis area to be deleted :').
    set_display_pos(5,2).  ed1 is []. option is '#fred deleting '.
    text('#fblue emphasis areas in the Database #fred          (Select the emphasis to be deleted)').
     Lb is list_box([?emphasisList],input_data,5,4,40,,,t,[list_select_event,double_click_event]).
     ed1 is edit_line(,,35,13,30).
     button (OK,OK,55,5,10). button(Cancel,can,55,9,10).
    end.
 end.

topic delete.
  get_data().
  set_title(?wmenu,'DELETE THE EMPHASIS AREA').
  FormatDisplay().
  set_display_pos(22,2). text(?ed2).
  set_display_pos(22,4). text(?ed1).
  set_display_pos(22,6). text(?ed3).
  set_display_pos(22,8). text(?ed4).
 set_display_pos(2,14). text('#fred Delete the emphasis area?').
  button(YES,yesdelete,28,14,10).  button(NO, notdelete,40,14,10).
   wait( ).
 end.

 topic yesdelete.
      emphasisField = remove(?emphasisField,[?ed]).
      emphasisList = remove(?emphasisList, get_text(?ed1)).
        written is 5.
```

```
        message is concat(get_text(?ed1),'#fblue   deletion has been done').
        disable_window(?wh).
        display_info().
 end.

 topic notdelete.
       message is ' '.  disable_window(?wh).  display_info().
 end.

 topic can.
 test is 0.
  selection().
 end.

topic &add.
     disable_menu_item(?m1,[E&xit]). text(#e). option is '#fred adding'.
    ed1 is []. ed2 is []. ed3 is []. ed4 is [].
    set_title(?wmenu,'ADD THE emphasis ').
     FormatDisplay().  set_display_pos(55,1).
     text('#fblue emphasis in Database     ').
     list_box(?emphasisList,,55,2,20,,,t).
     button ('Add data (OK)',Ok_Add,18,15,16).
     button (cancel, not_add, 39,15,8).
     set_focus(?ed2).

  topic not_add.
    editone is get_text(?ed1).  disable_window(?wh).
    message is ' '. display_info().
  end.

  topic Ok_Add.
    editone is get_text(?ed1).  disable_window(?wh).
     if one_of(?emphasisList, get_text(?ed1))
          then message is concat(?editone,'#fred   was in the emphasis file !#d')
     else  if ?editone is [] or get_text(?ed2) is [] then
                    message is '#fred Enter emphasis Number & emphasis Name Please !'
              else (edrec is combine(get_text(?ed2), get_text(?ed1),get_text(?ed3), get_text(?ed4)) and
                   emphasisField gets ([?edrec]) and
                    emphasisList gets (get_text(?ed1)) and
                       written is 5 and
                      message is concat(get_text(?ed1),'#fblue   has been added into a emphasis file')).
         display_info().

 end.
end.
topic FormatDisplay.
   wh is window(,8,8,77,17,,[Dialogwindow,visible]).
     text(#e). set_display_pos(1,2). disable_window(?wmenu).
     text('#fblue    Emphasis Number :

      Emphasis Name :

          Description    :
```

```
                              :#d ').
      if not(?option is '#fred deleting ') then
      ed1 is edit_line(?ed1,,22,4,30) and
      display_record().
end.

topic display_record.
      ed2 is edit_line(?ed2,,22,2,4).
      ed3 is edit_line(?ed3,,22,6,40).
      ed4 is edit_line(?ed4,,22,8,40).
end.

 topic &Help.
   wh is window(,8,8,74,16,,[Dialogwindow,visible]).
    text (#e,read ('info.hyp', concat ('//',emphasis) ,'//') ).
    set_file_pos ('info.hyp',0,beginning).
    b2 is button(Ok,continue,30,14).
    set_focus(?b2).
    wait().
    close('info.hyp'). close_window().
 end.

topic update_file.
   new_file('emphasis1.txt').
    write('emphasis1.txt',?emphasisList). close('emphasis1.txt').
   new_file('track.txt').
   write('track.txt',?emphasisfield).  close('track.txt').

end.
topic 'previous_menu'.
   if ?written is 5 then update_file().
   new_file('tempo.dat').
   write('tempo.dat',2).
  close_all().
  close_window(?wmenu).
    do(!main).
end.
topic 'quit_to_system'.
   if ?written is 5 then update_file().
   close_all().
   clear().
end.
(*----------END EMPHASIS.KB ----------*)

(*
      Program : elective.kb
      Author    : Suprapto *)

 wmenu is window(select:&quit,5,5,82,20,,[Popup,,ControlMenu,ThickFrame,ShowChildren,Siblings]).
 m1  is menu([[E&xit, 'Previous_menu', 'Quit_to_system'],&Help],select).
 trackel is read('trackelt.txt').
 aitrack is string_to_list(element(string_to_list(element(?trackel,1),'//'),1)).
```

```
 aielect is string_to_list(element(string_to_list(element(?trackel,1),'//'),2)).
 tstrack is string_to_list(element(string_to_list(element(?trackel,2),'//'),1)).
 tselect is string_to_list(element(string_to_list(element(?trackel,2),'//'),2)).
 setrack is string_to_list(element(string_to_list(element(?trackel,3),'//'),1)).
 seelect is string_to_list(element(string_to_list(element(?trackel,3),'//'),2)).
 dbtrack is string_to_list(element(string_to_list(element(?trackel,4),'//'),1)).
 dbelect is string_to_list(element(string_to_list(element(?trackel,4),'//'),2)).
 course1 is read('course1.txt'). close_all().
show_window(?wmenu). written is 0.

selection().

topic selection.
      text(#e).    set_title(?wmenu,' ').
      text ('#n#fred

                              MANAGEMENT  ELECTIVE DATA OBJECT#FBLUE
                              ----------#d').
      set_display_pos(16,15).
               text('#fblue          ----------').
      button('UPDATE THE ELECTIVE ',&update,25,6,30).
      button('DELETE THE ELECTIVE ',&delete,25,9,30).
      button('ADD THE ELECTIVE',&add,25,12,30).
      wait().
end.

topic E&xit.
      test is 0.
     selection().
end.

topic  select(item).
  do (?item).
end.


 topic &update.
    set_title(?wmenu,'UPDATE THE COURSE ELECTIVE').
     opt is '#fred updating'.
     ed1 is []. ed2 is []. option is 1. choice is 2. election is 3.
     FormatDisplay(?option,'updated !!').
     b is button (OK,OK,22,9,8).   button (Cancel,can,32,9,8).
    set_focus(?b).
  topic ok.
    option is 1.
     ed2 is [].
     second_display().
       b is button (OK,OKE,22,8,8).   button (Cancel,postpone,32,8,8).
  end.
  topic oke.
      ed2 is get_text(?ed2).
      do(concat('No',list_of_char(?ed1))).  close_window(?wh1). remove_topic(wh1).
      old is ?ed3.
```

```
      option is 2. choice is 5. second_display().
      button (OK,continue,22,12,8).
      button (cancel,postpone,32,12,8).
  end.
 topic postpone.
       message is ' '. display_info().
 end.
  topic No1.  if one_of(?aitrack, ?ed2) then ed3 is 'Required'  else ed3 is 'Optional'. end.
  topic No2.  if one_of(?tstrack, ?ed2) then ed3 is 'Required'  else ed3 is 'Optional'. end.
  topic No3.  if one_of(?setrack, ?ed2) then ed3 is 'Required'  else ed3 is 'Optional'. end.
  topic No4.  if one_of(?dbtrack, ?ed2) then ed3 is 'Required'  else ed3 is 'Optional'. end.

  topic continue.
     ed3 is get_text(?ed3).  ed2 is get_text(?ed2).
     do(concat('update', list_of_char(?ed1))).  written is 5.
     message is concat(?ed2,'#fblue  has already been updated !'). display_info().
  end.

  topic update1.
      if not (one_of(?ed3, ?old))
        then ( if one_of(?aitrack, ?ed2)
                    then  aitrack is remove(?aitrack, ?ed2) and aielect gets [?ed2]
               else  aielect is remove(?aielect, ?ed2) and aitrack gets [?ed2]).
  end.
  topic update2.
      if not (one_of(?ed3, ?old))
        then ( if one_of(?tstrack, ?ed2)
                    then  tstrack is remove(?tstrack, ?ed2) and tselect gets [?ed2]
               else  tselect is remove(?tselect, ?ed2) and tstrack gets [?ed2]).
  end.
 topic update3.
      if not (one_of(?ed3, ?old))
        then ( if one_of(?setrack, ?ed2)
                    then  setrack is remove(?setrack, ?ed2) and seelect gets [?ed2]
               else  seelect is remove(?seelect, ?ed2) and setrack gets [?ed2]).
  end.
 topic update4.
      if not (one_of(?ed3, ?old))
        then ( if one_of(?dbtrack, ?ed2)
                    then  dbtrack is remove(?dbtrack, ?ed2) and dbelect gets [?ed2]
               else  dbelect is remove(?dbelect, ?ed2) and dbtrack gets [?ed2]).
  end.

end.

 topic &delete.
    set_title(?wmenu,'DELETING THE COURSE ELECTIVE '). election is 2.
    ed1 is []. ed2 is []. option is 1. choice is 2. election is 2. opt is '#fred deleting'.
     FormatDisplay(?option,'deleted !!').
     b is button (OK,OK,22,9,8).  button (Cancel,can,32,9,8).
    set_focus(?b).
  topic ok.
     option is 1.
```

```
    ed2 is [].
    second_display().
    b is button (OK,OKE,22,8,8).   button (Cancel,postpone,32,8,8).
  end.
 topic postpone.
      message is ' '. display_info().
 end.
 topic oke.
     ed2 is get_text(?ed2).
    do(concat('No',list_of_char(?ed1))).  close_window(?wh1). remove_topic(wh1).
     option is 2. choice is 4. second_display().
     button ('Delete (OK)',continue,22,12,13).
     button (Cancel,not_delete,36,12,8).
 end.
  topic No1.  if one_of(?aitrack, ?ed2) then ed3 is 'Required'  else ed3 is 'Optional'. end.
  topic No2.  if one_of(?tstrack, ?ed2) then ed3 is 'Required'  else ed3 is 'Optional'. end.
  topic No3.  if one_of(?setrack, ?ed2) then ed3 is 'Required'  else ed3 is 'Optional'. end.
  topic No4.  if one_of(?dbtrack, ?ed2) then ed3 is 'Required'  else ed3 is 'Optional'. end.
  topic not_delete.
      message is ' '. display_info().
  end.
  topic continue.
    ed3 is get_text(?ed3). ed2 is get_text(?ed2).
    do(concat('delete',list_of_char(?ed1))). written is 5.
    message is concat(?ed2,'#fblue  has been deleted from Database'). display_info().
  end.
  topic delete1.  if one_of(?aitrack,?ed2) then aitrack is remove(?aitrack,?ed2)
                      else aielect is remove (?aielect,?ed2). end.
  topic delete2.  if one_of(?tstrack,?ed2) then tstrack is remove(?tstrack,?ed2)
                      else tselect is remove (?tselect,?ed2). end.
  topic delete3.  if one_of(?setrack,?ed2) then setrack is remove(?setrack,?ed2)
                      else seelect is remove (?seelect,?ed2). end.
  topic delete4.  if one_of(?dbtrack,?ed2) then dbtrack is remove(?dbtrack,?ed2)
                      else dbelect is remove (?dbelect,?ed2). end.
end.

topic can.
  close_window(). enable_window(?wmenu).
 selection().
end.
topic display_info.
   disable_window(?wmenu).
   disable_window(?wh).     disable_window(?wh1).
   wh2 is window(,19,10,54,12,,[Dialogwindow,visible]).
   text('#n #n    '). text(?message).
   text('#n #n #n Continue '). text(?opt). text(' the req track elective ?').
   button(Yes,yes,40,6,5).
   button(No,main_menu,47,6,5).
end.

topic yes.
      close_window(?wh1). remove_topic(wh1).
      close_window(?wh2). remove_topic(wh2).
```

```
      if exists(edit1) then ed1 is ?edit1.   if exists(edit2) then ed2 is ?edit2.
      enable_window(?wh). if one_of([2,3],?election) then choice is 2.
end.

topic main_menu.
      close_window(?wh). remove_topic(wh).
      close_window(?wh1). remove_topic(wh1).  close_window(?wh2). remove_topic(wh2).
      enable_window(?wmenu).
     selection().
end.

topic &add.
      set_title(?wmenu,'ADDING THE COURSE ELECTIVE').
      ed1 is []. ed2 is []. ed3 is [].  option is 2.  choice is 1. election is 1.
      FormatDisplay(?option,'added'). opt is 'adding'.
     button (OK,continue,22,9,5).
     button (cancel, can, 30 ,9,8).

  topic continue.
     second_display().
   button (OK,oke,22,12,5).
   button (cancel, not_add, 30 ,12,8).
 end.
  topic oke.
      ed2 is get_text(?ed2). ed3 is get_text(?ed3). written is 5.
    do(concat(&,?ed1)).  message is concat(?ed2,'#fblue  has been added into Database').
     ed2 is []. ed3 is []. display_info().
 end.
 topic not_add.
    message is ' '. display_info().
 end.

 topic &1. if ?ed3 is 'Required' then aitrack gets  (?ed2) else aielect gets  (?ed2). end.
 topic &2. if ?ed3 is 'Required' then  tstrack gets (?ed2) else tselect gets (?ed2). end.
 topic &3. if ?ed3 is 'Required' then setrack gets (?ed2) else seelect gets (?ed2). end.
 topic &4. if ?ed3 is 'Required' then dbtrack gets (?ed2) else  dbelect gets (?ed2). end.
end.

topic FormatDisplay(option,job).
   wh is window(,8,9,77,13,,[Dialogwindow,visible]). disable_window(?wmenu).
     text(#e).

     text('#n#fblue                    Select the elective track to be '). text(?job).

text('#n#n#fblue  Emphasis number :#d').

  radio_button([['1 - Artificial Intelligence',26,4],
                        ['2 - Tactical Computer System',26,5],
                        ['3 - Software Engineering',26,6],
                        ['4 - Military Data Processing',26,7]],track).

      ed1 is edit_line(?ed1,,22,4,3).
end.
```

```
topic second_display.
   disable_window(?wh).
   wh1 is window(,9,8,75,15,,[Dialogwindow,visible]).
  text(#e).  text('#fblue

  Emphasis number :


  Course number      :#d').

 set_display_pos(22,2). text(?datatrack).
     ed2 is edit_line(?ed2,,22,5,9).
     if ?option > 1
       then radio_button([[Required,33,9],[Optional,33,10]],elective) and
            set_display_pos(2,9) and
            text('#fblue Elective type          :  #d ') and
             ed3 is edit_line(?ed3,,22,9,10).
     set_display_pos(42,3).
     text('#fblue Req Track Elective').
     if ?ed1 is 1
        then tracklist is combine(?aitrack, ?aielect)
     else ( if ?ed1 is 2
                then tracklist is combine(?tstrack, ?tselect)
            else ( if ?ed1 is 3
                       then tracklist is ?setrack
                   else  tracklist is ?dbtrack)).
  list_box(?tracklist,input_data,45,4,10,,,,[list_select_event,double_click_event]).
  if ?election is 1 then set_display_pos(63,2) and text('#fblue Available') and
   set_display_pos(63,3) and text('#fblue  Courses') and
  list_box( remove(?course1,?tracklist),new_data,63,4,10,,,t,[list_select_event,double_click_event]).
end.

topic input_data(item).
  if  one_of([2,3],?choice)
    then  ed2 is ?item and ed2 is edit_line(?ed2,,22,5,9). edit2 is ?item.
end.

topic new_data(item).
  if ?choice is 1
    then ed2  is ?item and ed2 is edit_line(?ed2,,22,5,9).   edit1 is ?item.
end.

topic track(item).
      datatrack is ?item.
     char is list_of_char(?item). ed1 is first(?char).
     ed1 is edit_line(?ed1,,22,4,3).
     ed1 is get_text(?ed1). edit1 is ?ed1.
end.


topic elective(item).
  if one_of([1,5],?choice)
     then   ed3 is ?item and ed3 is edit_line(?ed3,,22,9,10).
```

```
end.

topic &Help.
  wh is window(,8,8,74,16,,[Dialogwindow,visible]).
   text (#e,read ('info.hyp', concat ('//',elective) ,'//') ).
   set_file_pos ('info.hyp',0,beginning).
   b2 is button(Ok,continue,30,14).
   set_focus(?b2).
    wait().
    close_window().
end.

topic update_file.
   trackel is [].
   trackel gets list_to_string(combine(list_to_string(?aitrack),list_to_string(?aielect)),'//').
   trackel gets list_to_string(combine(list_to_string(?tstrack),list_to_string(?tselect)),'//').
   trackel gets list_to_string(combine(list_to_string(?setrack),list_to_string(?seelect)),'//').
   trackel gets list_to_string(combine(list_to_string(?dbtrack),list_to_string(?dbelect)),'//').
(*  new_file('trackelt.txt').
   write('trackelt.txt',?trackel). close('trackelt.txt').*)
end.

topic 'previous_menu'.
  new_file('tempo.dat').
   write('tempo.dat',2).
   if ?written is 5 then  update_file(). close_all(). close_window(?wmenu).
    do(!main).
end.
topic 'quit_to_system'.
    if ?written is 5 then  update_file().
   close_all().
   clear().
end.
(*----------END ELECTIVE.KB ----------*)


(*       Title    :  course.kb
         Author : suprapto

*)

wmenu is window(select:&quit,5,5,82,21,,[Popup,,ControlMenu,ThickFrame,ShowChildren,Siblings]).
m1  is menu([[E&xit, 'Previous_menu', 'Quit_to_system'],&Help],select).
set_title(?wmenu,'COURSE OBJECT').

set_title(?wmenu,'COURSE OBJECT'). test is 0.
show_window(?wmenu). written is 0.
wh is window(,22,13,46,4,,[Dialogwindow,visible]).
    text('#n #fred                    Loading Data.....').
    use_font(system_font).
    make_modal(?wh). show_window(?wh).
read_file().  close_window().
```

```
topic read_file.
 course is read('course.txt').
 course1 is read('course1.txt').
 courseNum is string_to_list(element(string_to_list(?course1,'//'),1)).
 courseName is string_to_list(element(string_to_list(?course1,'//'),2),'@').
 length is list_length(?courseNum).  no is 1.
 repeat
    courseList gets concat(element(?courseNum,?no),' ',element(?courseName,?no)) and
    no is ?no + 1
 until ?no > ?length.
end.

selection().

topic selection.
      text(#e). enable_menu_item(?m1,[E&xit]).
      text ('#n#fred

                              MANAGEMENT  COURSE DATA OBJECT#FBLUE
                              ----------#d').
      set_display_pos(16,15).
               text('#fblue          ----------').
      button('UPDATE THE COURSE ',&update,24,6,32).
      button('DELETE THE COURSE ',&delete,24,9,32).
      button('ADD THE COURSE ',&add,24,12,32).
      wait().
end.

topic  select(item).

   do (?item).
end.

 topic &update.
    set_title(?wmenu, UPDATE THE COURSE').
    disable_menu_item(?m1,[E&xit]).  option is '#FRED updating'.
    displaybox().

topic displaybox.
       text(#e). set_display_pos(5,13). text('#fblue Course Number to be updated :').
    set_display_pos(5,2).  ed1 is [].
    text('#fblue COURSES in the Database #fred          (Select the course to be updated)').
     Lb is list_box([?courseList],input_data,5,4,40,,,t,[list_select_event,double_click_event]).
     ed1 is edit_line(,,35,13,10).
     button (OK,Ok,55,5,10). button(Cancel,can,55,9,10).
  end.
 end.

 topic input_data(item,info_name).
     edfield  is  ?item. ed1 is string_to_list(?edfield). ed1 is first(?ed1).
     ed1 is edit_line(?ed1,,35,13,10).
   if ?info_name  is  'double_click_event'  then ed1 is get_text(?ed1) and
         editone is ?ed1 and  check_data().
```

```
end.

topic Ok.
   if not(get_text(?ed1) is []) then ed1 is get_text(?ed1) and
         editone is ?ed1 and check_data()
     else  input_data(get_list_box(?Lb)).

end.

 topic check_data().
    if ?option is '#fred Updating' then edit_course()
      else ( if ?option is '#fred deleting ' then delete_course()
               else add_course()).
 end.

 topic edit_course.
    if   one_of(?courseNum, ?ed1)
      then edit()
    else
        message is concat(?ed1, '#fblue  was not found in the course file !') and
        display_info().
 end.

 topic delete_course.
    if   one_of(?courseNum, ?ed1)
      then delete()
    else
          message is concat(?ed1,'#fred  was not found in the course file !!') and
          display_info().
 end.

topic display_info.
   disable_window(?wmenu).
   disable_window(?wh).
   wh1 is window(,20,10,51,12,,[Dialogwindow,visible]).
   text('#n #n   '). text(?message).
   text('#n #n #n  Continue '). text(?option). text(' the course file ?').
   button(Yes,yes,35,6,6).
   button(No,main_menu,43,6,6).
end.

topic yes.
     close_window(?wh). remove_topic(wh).   enable_window(?wmenu).
      close_window(?wh1). remove_topic(wh1). ed1 is ?editone.
    if ?option is '#fred deleting '  then  ( if not(?message is ' ') then &delete()
                                                 else ed1 is edit_line(?ed1,,35,13,10))
       else ( if ?option is '#fred adding'  then  &add()
                else  ed1 is edit_line(?ed1,,35,13,10)).
end.

topic main_menu.
     close_window(?wh). remove_topic(wh).
     close_window(?wh1). remove_topic(wh1).
```

```
      enable_window(?wmenu).
     selection().
end.

topic get_data.
    ldrec  is element(?course, where(?courseNum, ?ed1)).
    ed is string_to_list(?ldrec,'//').
    ed2 is element(?courseName, where(?courseNum, ?ed1)).
    ed3 is first(string_to_list(element(?ed, 2))).
    edp is element(?ed,3).
    ed4 is last(string_to_list(element(?ed, 2))).  ed5 is element(?ed,4).
    ed6 is element(?ed, 5).
end.

topic edit.
    keyvalue is  ?ed1.
    get_data().
     set_title(?wmenu,'UPDATE THE COURSE').
     FormatDisplay().
     button (Update,yesUpdate,20,14,10).
     button (cancel, notUpdate, 32,14,10).

  topic notUpdate.
      message is ' '.  display_info().
  end.

  topic yesUpdate.
    ed1 is get_text(?ed1).
    ed2 is get_text(?ed2).
    ed3 is get_text(?ed3).
    ed4 is get_text(?ed4).
    ed5 is get_text(?ed5).
    edp is get_text(?edp).
    ed6 is get_text(?ed6).
    set_display_pos(20,16).
    disable_window(?wh).
    if one_of(?courseNum, ?ed1)
      then (if ?ed1 is ?keyvalue
                then  updatedata()
            else message is concat(?ed1,'#fred   was in the course file !#d')
                  and display_info())
    else message is concat(?ed1,'#Fred   was not in the course file !  Please, add first') and
         display_info().
  end.
end.

topic updatedata.
     edrec is  list_to_string(combine(?ed1,concat(?ed3,' ',?ed4),?edp,?ed5,?ed6),'//').
     course is replace(?course, ?ldrec,?edrec).
     courseName is replace(?courseName, element(?courseName,
                          where(?courseNum,?ed1)), ?ed2).
     ed is concat(?ed1,' '). ed is concat(?ed,?ed2).
     courseList is replace(?courseList,?edfield,?ed).
```

```
    written is 5.
    message is concat(?ed1,'#fblue   has already been updated !!#d').
    display_info().
end.

topic &delete.
   disable_menu_item(?m1,[E&xit]).
   set_title(?wmenu,'DELETE THE COURSE').
   displaybox().

topic displaybox.
      text(#e). set_display_pos(5,13). text('#fblue Course Number to be deleted :').
   set_display_pos(5,2).  ed1 is [].  option is '#fred deleting '.
   text('#fblue COURSES in the Database #fred          (Select the course to be deleted)').
    Lb is list_box([?courseList],input_data,5,4,40,,,t,[list_select_event,double_click_event]).
    ed1 is edit_line(,,35,13,10).
    button (OK,OK,55,5,10). button(Cancel,can,55,9,10).
   end.
end.

topic delete.
  get_data().
  set_title(?wmenu,'DELETE THE COURSE').
  FormatDisplay().
  set_display_pos(20,2). text(?ed1).
  set_display_pos(20,4). text(?ed2).
  set_display_pos(20,6). text(?ed3). text(' / '). text(?ed4).
  set_display_pos(20,8). text(?edp).
  set_display_pos(20,10). text(?ed5).
  set_display_pos(20,11). text(?ed6).
 set_display_pos(4,14). text('#fred Delete this Course ?').
  button(YES,yesdelete,24,14,10).  button(NO, notdelete,35,14,10).
   wait( ).
end.

topic yesdelete.
     course = remove(?course,?ldrec).
     courseList = remove(?courseList,?edfield).
     courseNum = remove(?courseNum, ?ed1).
     courseName = remove(?courseName, ?ed2).
     written is 5.
      message is concat(?ed1,'#fblue   deletion has been done').
      display_info().
end.

topic notdelete.
     message is ' '. display_info().
end.

topic can.
test is 0.
 selection().
end.
```

```
topic &add.
     disable_menu_item(?m1,[E&xit]). text(#e). option is '#fred adding'.
    ed1 is []. ed2 is []. ed3 is []. ed4 is []. edp is []. ed5 is []. ed6 is [].
    set_title(?wmenu,'ADD THE COURSE ').
     FormatDisplay(). set_display_pos(55,1).
     text('#fblue Courses in Database    ').
     list_box(?CourseList,,51,2,25,,,t).
     button ('Add data (OK)',Ok_Add,18,15,16).
     button (cancel, not_add, 39,15,8).
     set_focus(?ed1).

  topic not_add.
    editone is get_text(?ed1). disable_window(?wh).
    message is ' '. display_info().
  end.

  topic Ok_Add.
    editone is get_text(?ed1). disable_window(?wh).
     if one_of(?courseNum, get_text(?ed1))
          then message is concat(?editone,'#fred   was in the course file !#d')
     else  if (?editone is [] or get_text(?ed2) is []) then message is '#fred Enter Course Number & Course Name Please
!'
             else (edrec is combine(get_text(?ed1),  concat(get_text(?ed3),' ', get_text(?ed4)),
                                          get_text(?edp), get_text(?ed5),get_text(?ed6)) and
                   course gets (list_to_string(?edrec,'//')) and courseName gets get_text(?ed2) and
                   ed is concat(get_text(?ed1),' ',get_text(?ed2))
                    and  courseNum gets (get_text(?ed1)) and courseList gets (?ed) and
                      set_display_pos(50,15) and written is 5 and
                      message is concat(get_text(?ed1),'#fblue   has been added into a course file')).
         display_info().

 end.
end.
topic FormatDisplay.
   wh is window(,17,8,65,17,,[Dialogwindow,visible]).
     text(#e). set_display_pos(1,2). disable_window(?wmenu).
     text('#fblue     Course Number :

      Course Name :

     Credit Lec/Lab :

      Prerequisite   :

      Description     :#d ').
      if not(?option is '#fred deleting ') then
      ed1 is edit_line(?ed1,,20,2,8) and
      display_record().
end.

topic display_record.
      ed2 is edit_line(?ed2,,20,4,30).
      ed3 is edit_line(?ed3,,20,6,4).
```

```
      set_display_pos(25,6). text('/').
      ed4 is edit_line(?ed4,,27,6,4).
      edp is edit_line(?edp,,20,8,25).
      ed5 is edit_line(?ed5,,20,10,40).
      ed6 is edit_line(?ed6,,20,12,40).
end.

 topic &Help.
   wh is window(,8,8,74,16,,[Dialogwindow,visible]).
    text (#e,read ('info.hyp', concat ('//',course) ,'//') ).
    set_file_pos ('info.hyp',0,beginning).
    b2 is button(Ok,continue,30,14).
     topic continue.
        close('info.hyp'). close_window().
     end.
 end.

topic update_file.
   new_file('course1.txt').
    write('course1.txt',list_to_string(combine(list_to_string(?courseNum,' '),
                list_to_string(?courseName,'@')),'//')).
    close('course1.txt').
   new_file('course.txt').
   write('course.txt',?course).  close('course.txt').
  close_all().
end.

topic 'previous_menu'.
   new_file('tempo.dat').
   write('tempo.dat',2).
   if ?written is 5 then update_file().
   close_window().
    do(!main).
end.
topic 'quit_to_system'.
   if ?written is 5 then update_file().
   close_all().
   clear().
end.
(*-------------END COURSE.KB ----------*)


 (*     Program        : quarter.kb
       Programmer : suprapto
*)
 wmenu is window(select:&quit,5,5,82,20,,[Popup,,ControlMenu,ThickFrame,ShowChildren,Siblings]).
m1  is menu([[E&xit, 'Previous_menu', 'Quit_to_system'],&Help],select).
set_title(?wmenu,'QUARTER OFFERING OBJECT'). written is 0.
show_window(?wmenu).  qtrName is [Winter,Spring,Summer,Fall].
                              qtrList is [1,2,3,4].
message is read('QUARTER.TXT').
(* read file quarter.txt (quarter#, year, {course#})
   It contains courses to be offered in each quarter *)
```

```
selection().

topic select(option).
    do(?option).
end.
topic selection.
     text(#e). text ('#n#fred

                        MANAGEMENT  QUARTER OFFERING OBJECT#FBLUE
                        ----------------------------------------------------------------------#d').
     set_display_pos(20,15).
     text('#fblue ----------------------------------------------------------------').
     button('UPDATE  THE QUARTER OFFERING ',&update,20,6,42).
     button('DELETE THE QUARTER OFFERING ',&delete,20,9,42).
     button('ADD A NEW QUARTER OFFERING ',&add,20,12,42).
     wait().
end.

topic &add.
    set_title(?wmenu,' ADD A NEW RECORD  QUARTER OFFERING').
    ed1 is []. ed2 is []. ed3 is []. opt is '#fred adding'.
    display().
end.


topic display.
    wh is window(,7,8,78,16,,[Dialogwindow,visible]).
    use_font(system_font).
   make_modal(?wh).
    year is 0. quarter is 0. ed1 is []. ed2 is [].
   text(#e).  set_display_pos(26,2). text('#fblue Select the quarter to be added').
   set_display_pos(26,3).                text('#fred ----------------------------------------#d').
    radio_button([[Winter,38,6],[Spring,50,6],
                  [Summer,38,7],[Fall,50,7]], quarter).
    radio_button([[91,33,9],[92,42,9],[93,51,9]],year).
    ed1 is edit_line(?ed1,,27,6,9).
    ed2 is edit_line(?ed2,,27,9,4).
   set_display_pos(0,5).
   text('#fblue

          Quarter Name    :


           Y e a r              :#d').
    button(Ok,ok_add,28,12,6).   button(cancel,cancellation,37,12,8).
show_window(?wh).
end.

topic ok_add.  checkdatas(). end.

topic checkdatas().
   search_data().
    listboxformat().
```

```
end.

topic listboxformat.
      year is 0. quarter is 0.
     text(#e).  set_display_pos(24,2). text('Enter the course to be added').
          set_display_pos(24,3). text('#fred ----------------------------------------------#d').
     set_display_pos(12,6). text('#fblue Quarter Name  :  '). text(?ed1).
     set_display_pos(12,9). text('#fblue Academic year :  '). text(?ed2).
     set_display_pos(55,4).
  set_display_pos(55,4).
   text('#fblue Courses are offered').
   set_display_pos(60,5).
    text('in '). text(?edit1). text('  '). text(?edit2).
    Listbox is list_box([?recordNum],,60,6,10,,,t).
   set_display_pos(10,12). text('#fblue   Course Number :').
  ed3 is edit_line(?ed3,,29,12,10).
    set_display_pos(9,14).
    text('#fblue Add this course ? '). button(Yes,yesadd,27,14,7).
    button(No,not_add,37,14,6).
end.
topic continueAdd.
     close_window(). ed3 is []. listboxformat().
end.
topic not_add.
    info is ' '.  display_info().
    button(Yes,continueAdd, 42,4,5). button(No,MainMenu,49,4,5).
    show_window().
end.

topic yesAdd.
      ed3 is get_text(?ed3).
     if ?ed3 is []  then  info is '#  #fred    Select course number, please !'  and
                test is 3 and disp  _message()
     else (if one_of(?recordNum, ?ed3) then
           info is concat(?ed3,'#fred  was  in the database !')
           else AddRecord() and  written is 5 and
                 info is concat(?ed3,'#fblue  has been added into Database')) and
          display_info() and button(Yes,continueAdd, 42,4,5) and button(No,MainMenu,49,4,5)
          and show_window().
end.
topic display_info.
       w2 is window(,16,12,56,8,,[Dialogwindow,visible]).
       text('#n    '). text(?info). text('#n #n  ').
       text(' Continue '). text(?opt). text(' the quarter offering  ?').
      use_font(system_font).
     make_modal(?w2).
end.


topic addrecord.
    recordNum gets (?ed3).
    newRec is concat(?Year_Qtr, '//', list_to_string(?recordNum,' ')).
    message is replace(?message, ?oldRec, ?newRec).
```

```
    oldRec is ?newRec.
end.
(*-------------------------------------------------------------------------*)

topic &update.
    set_title(?wmenu,'UPDATE A QUARTER OFFERING').
    ed1 is []. ed2 is []. ed3 is []. opt is '#fred updating'.
     header = '#fblue Select the quarter to bo updated#d'.
     Format_Display().
 end.

topic format_display.

     wh is window(,7,8,78,16,,[Dialogwindow,visible]).
     use_font(system_font).
    make_modal(?wh).
     year is 0. quarter is 0. ed1 is []. ed2 is [].
    text(#e).  set_display_pos(26,2). text(?header).
    set_display_pos(26,3). text('#fred ----------#d').
     radio_button([[Winter,38,6],[Spring,50,6],
                   [Summer,38,7],[Fall,50,7]], quarter).
     radio_button([[91,33,9],[92,42,9],[93,51,9]],year).
     ed1 is edit_line(?ed1,,27,6,9).
     ed2 is edit_line(?ed2,,27,9,4).
    set_display_pos(0,5).
    text('#fblue

            Quarter Name     :


             Y e a r              :#d').
    button(Ok,ok_update,28,12,6).   button(cancel,cancellation,37,12,8).
 show_window(?wh).
end.

topic cancellation.
      info is [].
     display_info().       button(Yes,continue, 42,4,5).   button(No,main_menu,49,4,5).
     show_window().
     topic continue. close_window(). end.
     topic main_menu. close_window(). close_window(). end.
end.

topic ok_update.
  if ( not(?quarter is 0) and not(?year is 0)) then  search_data() and ed3 is [] and display_box()
     else info is '#n  #fred            Please select the options  ' and test is 0 and
          display_message().
end.


topic search_data.
  no is 1. recordNum is []. totalrec is list_length(?message).
     repeat
```

```
        oldrec is element(?message,?no) and
        record is string_to_list(?oldrec,'//') and
        ( if first(?record) is concat(?ed2,' ',element(?qtrList, where(?qtrName,?ed1)))
            then Year_Qtr is first(?record) and
                    recordNum is string_to_list(rest(?record),' ') and no is ?totalrec) and
         no = ?no + 1
    until ?No > ?totalrec.

end.


topic display_box.
    year is 0. quarter is 0.  edit3 is []. ed3 is [].
     text(#e).  set_display_pos(15,2).
          text('Select a course number to be updated from #fred the list box ').
           set_display_pos(15,3).
          text('#fred----------------------------------------------------------------------------------------#d').
     set_display_pos(12,6). text('#fblue Quarter Name  :  '). text(?ed1).
     set_display_pos(12,8). text('#fblue Academic year :  '). text(?ed2).
     set_display_pos(55,4).
    text('#fblue Courses offered').
   set_display_pos(55,5). text('in ').
   text(?edit1). text(' ').  text(?edit2).
   Lb is list_box([?recordNum],inputdatas,55,6,12,,,t,list_select_event).
     set_display_pos(10,10). text('#fblue   Previous Course Number :').
     set_display_pos(10,12). text('#fblue          New Course Number  :').
     ed3 is edit_line(?ed3,,38,12,10).
     set_display_pos(9,14).
     text('#fblue Update this course ? '). button(Yes,yesUpdate,29,14,6).
     button(Cancel,not_Update,38,14,8).
end.


topic not_update.
    info is ' '.  display_info().
    button(Yes,continueUpdate, 42,4,5). button(No,MainMenu,49,4,5).
    show_window().
end.

topic inputdatas(item).
     edit3 is ?item.  set_display_pos(38,10). text(?edit3).
end.

topic display_message.
             wh is window(,28,12,40,6,,[Dialogwindow,visible]).
             text(?info).
             button(Continue,con,13,4,15).
            use_font(system_font).
            make_modal(?wh). show_window().
            topic con. close_window().
               if ?test is 1 then display_box()
                   else if ?test is 2 then formatdisplays()
                           else if ?test is 3 then listboxformat().
```

```
        end.
end.

topic yesUpdate.
     ed3 is get_text(?ed3).
    if ?ed3 is [] then  info is '#n #fred    Select course number, please !' and
              test is 1 and display_message()
    else (if one_of(?recordNum, ?ed3) then
          info is concat(?ed3,'#fred  was  in the database !')
          else updaterecord() and  written is 5 and
                info is concat(?edit3,'#fblue  has been changed with ',?ed3,' from the database')) and
        display_info() and button(Yes,continueUpdate, 42,4,5) and button(No,MainMenu,49,4,5)
        and show_window().
end.
topic continueUpdate.
  close_window(). ed3 is []. display_box().
end.
topic MainMenu.
 close_window(). close_window().
end.

topic updaterecord.
   recordNum is replace(?recordNum,?edit3,?ed3).
   newRec is concat(?Year_Qtr,'//',list_to_string(?recordNum,' ')).
   message is replace(?message, ?oldRec, ?newRec).
    oldRec is ?newRec.
end.


(*=========================================*)

topic &delete.
    set_title(?wmenu,'DELETE A RECORD OF QUARTER OFFERING').
    ed1 is []. ed2 is []. ed3 is []. opt is '#fred deleting'. recordNum is [].
     header = '#fblue Select the quarter to be deleted#d'.
     FormDisplay().
end.

topic Formdisplay.
     formatdisplays().
     if ?recordNum is [] then
     button(Ok,ok_delete,28,12,6) and   button(cancel,cancellation,37,12,8).
end.
topic formatdisplays.
 disable_window(?wmenu).
     wh is window(,7,8,78,16,,[Dialogwindow,visible]).
     year is 0. quarter is 0.
    text(#e).  set_display_pos(26,2). text(?header).
    set_display_pos(26,3). text('#fred ----------#d').
     radio_button([[Winter,38,6],[Spring,50,6],
                   [Summer,38,7],[Fall,50,7]], quarter).
     radio_button([[91,33,9],[92,42,9],[93,51,9]],year).
     ed1 is edit_line(?ed1,,27,6,9).
```

```
    ed2 is edit_line(?ed2,,27,9,4).
   set_display_pos(0,5).
   text('#fblue

            Quarter Name     :


             Y e a r              :#d').
end.

topic ok_delete.
 if ( not(?quarter is 0) and not(?year is 0)) then  search_data() and ed3 is []
          and displaybox()
   else info is '#n  #fred          Please select the options  ' and test is 0 and
        display_message().
end.

topic year(item).
  ed2 is ?item. year is 1.
  ed2 is edit_line(?ed2,,27,9,4). ed2 is ?item. edit2 is ?item.
end.

topic quarter(item).
    ed1 is ?item. quarter is 1.
   ed1 is edit_line(?ed1,,27,6,9).   ed1 is ?item. edit1 is ?item.
end.

topic displaybox.
    year is 0. quarter is 0.
   text(#e).  set_display_pos(26,2). text('Select the courses to be deleted').
       set_display_pos(26,3). text('#fred ----------#d').
   set_display_pos(12,6). text('#fblue Quarter Name  :  '). text(?ed1).
   set_display_pos(12,9). text('#fblue Academic year :  '). text(?ed2).
   set_display_pos(55,4).
   text('#fblue Courses are offered').
   set_display_pos(55,5). text('in ').
   text(?edit1). text(' ').  text(?edit2).
   Lb is list_box([?recordNum],inputdata,55,6,12,,t,t,list_select_event).
   set_display_pos(10,12). text('#fblue   Course Number :').
   ed3 is edit_line(?ed3,,29,12,25).
   set_display_pos(9,14).
   text('#fblue Delete this course ? '). button(Yes,yesdelete,29,14,6).
   button(Cancel,not_delete,38,14,8).
end.
topic inputdata(item).
   ed3 is ?item. edit3 is ?item. ed3 is edit_line(?ed3,,29,12,25).
end.

topic not_delete.
   info is ' '.  display_info().
   button(Yes,continueDelete, 42,4,5). button(No,MainMenu,49,4,5).
   show_window().
end.
```

```
topic continueDelete.
     close_window(). ed3 is []. displaybox().
end.



topic yesDelete.
     ed3 is get_text(?ed3).

    if ?ed3 is [] then  info is '#n #fred    Select course number, please !'  and
              test is 2 and display_message()
    else (if not(intersect(?recordNum, ?edit3) is ?edit3) then
          info is concat(?ed3,'#fred  was not  in the database !')
          else removerecord() and  written is 5 and
                info is concat(?ed3,'#fblue  has been deleted from Database')) and
        display_info() and button(Yes,continueDelete, 42,4,5) and button(No,MainMenu,49,4,5)
        and show_window().
end.
topic continueDelete.
  close_window(). ed3 is []. displaybox().
end.

topic removerecord.
   recordNum is remove(?recordNum,?edit3).
   newRec is concat(?Year_Qtr,'//', list_to_string(?recordNum,' ')).
   message is replace(?message, ?oldRec, ?newRec).
    oldRec is ?newRec.
end.

topic &Help.
  wh is window(,8,8,74,16,,[Dialogwindow,visible]).
   text (#e,read ('info.hyp', concat ('//',quarter) ,'//') ).
   set_file_pos ('info.hyp',0,beginning).
   b2 is button(Ok,keep_on,30,14).
   set_focus(?b2).
    wait().
  topic keep_on.
   close('info.hyp'). close_window().
  end.
end.

topic update_file.
 new_file('quarter.txt').
 write('quarter.txt',?message).
 close('quarter.txt').
end.

topic 'previous_menu'.
 if ?written is 5 then update_file().
   new_file('tempo.dat'). write('tempo.dat',2).
    close_all(). close_window(?wmenu).
   do(!main).
end.
```

```
topic 'quit_to_system'.
    if ?written is 5 then update_file().
  close_all(). close_window(?menu).
  clear().
end.
(*----------END QUARTER.KB----------*)

(*      Title   : STUDENT.KB
        Author : suprapto

*)

wmenu is window(select:&quit,5,4,82,23,,[Popup,,ControlMenu,ThickFrame,ShowChildren,Siblings]).
m1  is menu([[E&xit, 'Previous_menu', 'Quit_to_system'],&Help],select).
set_title(?wmenu,'STUDENT  OBJECT').
qtrList is ['91 1','91 2','91 3','91 4','92 1','92 2','92 3','92 4','93 1','93 2','93 3','93 4'].
quarterName is ['Winter 91','Spring 91','Summer 91','Fall 91','Winter 92','Spring 92',
                'Summer 92','Fall 92','Winter 93','Spring 93','Summer 93','Fall 93'].
trackList is ['Artificial Intelligence','Tactical Comp Systems','Software Engineering',
          'Military Data Processing']. trackNumber is [1,2,3,4].
quarterNumber is [1,2,3,4,5,6,7,8,9,10,11,12].
qtrNumber is [1,2,3,4,5,6,7,8].
qtrName is ['Winter 91','Spring 91','Summer 91','Fall 91',
            'Winter 92','Spring 92','Summer 92','Fall 92'].
set_title(?wmenu,'STUDENT OBJECT'). test is 0.
show_window(?wmenu). written is 0.

read_file().
topic read_file.
SList is read('student1.txt').

 Slength is list_length(?SList).  No is 1.  Sdt_Id is [].
  repeat
    Sdt_Id gets(first(string_to_list(element(?SList,?No))))  and  No = ?No + 1
  until ?no > ?Slength.


SInfo is read('student.txt').
(* read file student.txt that contains student_Id, name (kept in SList) and
            student Information(name,rank,section etc) kept in SInfo  *)


(* course1 is read('course1.txt').
 courseNum is string_to_list(element(string_to_list(?course1,'//'),1)).
 courseName is string_to_list(element(string_to_list(?course1,'//'),2),'@').  *)
end.

selection().

topic selection.
      text(#e). enable_menu_item(?m1,[E&xit]).
      text ('#n#fred
```

```
                    MANAGEMENT  COURSE DATA OBJECT#FBLUE
                    ----------------------------------------------------------#d').
    set_display_pos(16,17).
            text('#fblue          ----------------------------------------------------------').
    button('UPDATE THE STUDENT'S INFO',&update,24,7,32).
    button('DELETE THE STUDENT'S INFO',&delete,24,10,32).
    button('ADD THE STUDENT'S INFO',&add,24,13,32).
    wait().
end.

topic  select(item).

  do (?item).
end.

topic &update.
   set_title(?wmenu,'UPDATE THE COURSE').
   disable_menu_item(?m1,[E&xit]).  option is '#FRED updating'.
    wh is window(,7,7,79,19,,[Dialogwindow,visible]).
    use_font(system_font).
   make_modal(?wh).
   displaybox().

topic displaybox.
      text(#e). set_display_pos(5,13). text('#fblue Student's name to be updated :').
   set_display_pos(5,2).  ed1 is [].
   text('#fblue STUDENT in the Database #fred          (Select the student to be updated)').
    Lb is list_box([?SList],input_data,5,4,30,,,t,[list_select_event,double_click_event]).
    ed1 is edit_line(,,35,13,15).
    button (OK,Ok,55,5,10). button(Cancel,can,55,9,10).
  end.
end.

topic can. close_window(). enable_menu_item(?m1,[E&xit]). end.

topic input_data(item,info_name).
    edfield  is  ?item.   SKey is ?item. ed1 is string_to_list(?edfield).
    ed1 is first(?ed1).  keyValue is ?ed1.
    ed1 is edit_line(?ed1,,35,13,15).

  if ?info_name  is  'double_click_event'  then ed1 is get_text(?ed1) and
        editone is ?ed1 and  check_data().
end.

topic Ok.
   if not(get_text(?ed1) is []) then ed1 is get_text(?ed1) and
          editone is ?ed1 and check_data()
     else  input_data(get_list_box(?Lb)).

end.

  topic check_data().
    if ?option is '#fred Updating' then edit_student()
```

```
      else ( if ?option is '#fred deleting ' then delete_student()
               else add_student()).
 end.

 topic edit_student.
    if   one_of(?SList, ?skey)
      then edit()
    else
        message is concat(?ed1, '#fblue  was not in the student file !') and
        display_info().
 end.

 topic delete_student.
    if   one_of(?courseNum, ?ed1)
      then delete()
    else
          message is concat(?ed1,'#fred  was not  in the student file !!') and
          display_info().
 end.
topic display_info.
     w2 is window(,16,12,56,8,,[Dialogwindow,visible]).
     text('#n   '). text(?info). text('#n #n  ').
     text(' Continue '). text(?option). text(' the student's info  ?').
    use_font(system_font).
   make_modal(?w2).
end.

topic get_data.
 ed1 is string_to_list(?skey).  ed2 is concat(element(?ed1,2),' ',element(?ed1,3)).
 ed1 is first(?ed1).
 SField is element(?SInfo, where(?SList,?Skey)).
 student_info is string_to_list(?SField,'//').
 ed4 is element(?student_info,3). ed3 is element(?student_info,2).
 ed5 is (element(?student_info,5)).
 ed6 is (element(?student_info,6)).
 ed7 is (element(?student_info,7)).
 ed8 is (element(?student_info,8)).
 ed9 is element(?student_info,4).
end.

topic FormatDisplay.
  wh is window(,8,8,77,17,,[Dialogwindow,visible]).
   text(#e). set_display_pos(1,2). disable_window(?wmenu).
   text('#fblue  Student _Id  :                          N a m e  :

    R a n k   :                      4th quarter  :
                                       8th quarter  :
     Section :

     Track Name  :

Selected Courses :
```

```
 Taken Courses     : #d ').
        list_box([?qtrName],input_track,60,4,13,2,,,list_select_event).
       list_box([?trackList],input_quater,45,7,26,2,,,list_select_event).
     display_record().
end.
topic input_track(item).
  if not(?option is '#fred deleting') then get_trackName(?item).
end.
topic get_trackName(item).
     ed5 is element(?quarterNumber, where(?quarterName, ?item)).
     ed6 is ?ed5+3.
    set_display_pos(45,4). text('                                    ').
    set_display_pos(45,5). text('                                    ').
    set_display_pos(45,4). text(element(?quarterName, where(?quarterNumber, ?ed5))).
    set_display_pos(45,5). text(element(?quarterName, where(?quarterNumber, ?ed6))).
    ed5 is element(?qtrList, where(?quarterNumber,?ed5)).
    ed6 is element(?qtrList, where(?quarterNumber,?ed6)).
end.
topic input_quater(item).
 if not(?option is '#fred deleting') then
    ed9 is element(?trackNumber, where(?trackList, ?item)) and
    set_display_pos(20,8) and text('                                                ') and
    set_display_pos(20,8) and text(?item).
end.
topic display_record.
     ed1 is edit_line(?ed1,,15,2,12).
     ed2 is edit_line(?ed2,,45,2,20).
     ed3 is edit_line(?ed3,,15,4,10).
    set_display_pos(45,4). text(element(?qtrName, where(?qtrList, ?ed5))).
    set_display_pos(20,8). text(element(?trackList, where(?trackNumber, ?ed9))).
     ed4 is edit_line(?ed4,,15,6,8).
     set_display_pos(45,5). text(element(?qtrName, where(?qtrList, ?ed6))).
     ed7 is edit_line(?ed7,,20,10,52).
     ed8 is edit_line(?ed8,,20,12,52).
end.
topic edit.
    keyvalue is  ?ed1.
    get_data().
     set_title(?wmenu,'UPDATE THE STUDENT'S INFO').
     FormatDisplay().
     button (Update,yesUpdate,20,14,10).
     button (cancel,cancellation, 32,14,10).
end.

  topic cancellation.
     info is [].
     display_info().      button(Yes,continue, 42,4,5).   button(No,mainmenu,49,4,5).
     show_window().
     topic continue.
          close_window().
          if not(?option is '#fred adding') then close_window().
    end.
     topic mainmenu.
```

```
        close_window(). close_window(). enable_menu_item(?m1,[E&xit]).
        if not(?option is '#fred adding') then  close_window().
    end.
end.

 topic yesUpdate.
   ed1 is get_text(?ed1).
   ed2 is get_text(?ed2).
   ed3 is get_text(?ed3).
   ed4 is get_text(?ed4).
   ed8 is get_text(?ed8).
   ed7 is get_text(?ed7).
   set_display_pos(20,16). newKey is list_to_string(combine(?ed1,?ed2),' ').

  if not(one_of(?Slist, ?newkey))
        then ( if one_of(different(?Sdt_Id,?keyValue),?ed1)
                  then  info is  concat(?ed1,'#fred   was in the course file !#d')
                       else  updatedata())
      else (if ?ed1 is ?keyValue
                then  updatedata()
             else info is concat(?ed1,'#fred   was in the course file !#d')).
           display_info().  button(Yes,continueUpdate, 42,4,5).  button(No,MainMenu,49,4,5).
         show_window().
 end.

topic continueUpdate. close_window(). close_window(). close_window(). &update(). end.

topic MainMenu.
    close_window(). close_window(). enable_menu_item(?m1,[E&xit]).
    close_window().
end.

topic updatedata.
     Slist is replace(?SList,?Skey,?newKey).
     edrec is  list_to_string(combine(?ed1,?ed3,?ed4,?ed9,?ed5,?ed6,?ed7,?ed8),'//').
    Sinfo is replace(?Sinfo, ?SField,?edrec).
     written is 5.
     info is concat(?ed1,'#fblue   has already been updated !!#d').
end.

topic &delete.
    disable_menu_item(?m1,[E&xit]).
    set_title(?wmenu,'DELETE THE STUDENT'S INFO').
    wh is window(,7,7,79,19,,[Dialogwindow,visible]).
     use_font(system_font).
    make_modal(?wh).
    displaybox().

topic displaybox.
       text(#e). set_display_pos(5,13). text('#fblue The student's Id to be deleted :').
    set_display_pos(5,2).  ed1 is []. option is '#fred deleting'.
    text('#fblue STUDENT in the Database #fred          (Select the student to be deleted)').
     Lb is list_box([?SList],input_data,5,4,30,,,t,[list_select_event,double_click_event]).
```

```
    ed1 is edit_line(,,35,13,15).
    button (OK,delete,55,5,10). button(Cancel,can,55,9,10).
   end.
end.

topic delete.
  get_data().
  set_title(?wmenu,'DELETE THE STUDENT'S INFO').
  FormatDisplay().
  set_display_pos(4,14). text('#fred Delete this record ?').
  button(YES,yesdelete,24,14,10).  button(NO, cancellation,35,14,10).
   wait( ).
end.
topic yesdelete.
     Slist is remove(?SList,?Skey).
    Sinfo is remove(?Sinfo, ?SField).
     written is 5.
     info is concat(?ed1,'#fblue   has already been updated !!#d').
           display_info().  button(Yes,continuedelete, 42,4,5).  button(No,MainMenu,49,4,5).
         show_window().
end.

topic continueUpdate. close_window(). close_window(). close_window(). &delete(). end.


topic &add.
     disable_menu_item(?m1,[E&xit]).  option is '#fred adding'.
    ed1 is []. ed2 is []. ed3 is []. ed4 is []. ed7 is []. ed5 is []. ed6 is []. ed8 is []. ed9 is [].
    set_title(?wmenu,'ADD THE STUDENT'S INFO').
     FormatDisplay().  set_display_pos(55,1).
     button ('Add data (OK)',Ok_Add,18,15,16).
     button (cancel, cancellation, 39,15,8).
     set_focus(?ed1).

 topic Ok_Add.
  ed1 is get_text(?ed1).
   ed2 is get_text(?ed2).
   ed3 is get_text(?ed3).
   ed4 is get_text(?ed4).
   ed8 is get_text(?ed8).
   ed7 is get_text(?ed7).
     newKey is list_to_string(combine(?ed1,?ed2),' ').
     if not(one_of(?Slist, ?newkey))
        then  add_record()
     else info is concat(?ed1,'#fred   was in the course file !#d').
           display_info().  button(Yes,continue_add, 42,4,5).  button(No,Main_Menu,49,4,5).
         show_window().
 end.
 topic continue_add.  close_window().  close_window().  &add().  end.
 topic Main_menu.
      enable_menu_item(?m1,[E&xit]). close_window().
      close_window().
end.
```

```
 topic add_record().
     Slist gets(?newKey).
     edrec is  list_to_string(combine(?ed1,?ed3,?ed4,?ed9,?ed5,?ed6,?ed7,?ed8),'//').
    Sinfo gets(?edrec).
     written is 5.
     info is concat(?ed1,'#fblue   has already been updated !!#d').
 end.
end.


 topic &Help.
   wh is window(,8,8,74,16,,[Dialogwindow,visible]).
    text (#e,read ('info.hyp', concat ('//',student) ,'//') ).
    set_file_pos ('info.hyp',0,beginning).
    b2 is button(Ok,continue,30,14).
   topic continue.
      close('info.hyp'). close_window().
    end.
 end.

topic update_file.
   new_file('course1.txt').
    write('course1.txt',list_to_string(combine(list_to_string(?courseNum),
                list_to_string(?courseName,'@')),'//')).
    close('course1.txt').
   new_file('course.txt').
   write('course.txt',?course).  close('course.txt').

  close_all().
end.

topic 'previous_menu'.
   new_file('tempo.dat').
   write('tempo.dat',2).
   if ?written is 5 then update_file().
   close_window().
    do(!main).
end.
topic 'quit_to_system'.
   if ?written is 5 then update_file().
   close_all().
   clear().
end.
(*-------------END STUDENT.KB-----------*)

(*      program    : emphasis.kb
        author      : suprapto
*)
wmenu is window(select:&quit,5,5,82,21,,[Popup,,ControlMenu,ThickFrame,ShowChildren,Siblings]).
m1  is menu([[E&xit, 'Previous_menu', 'Quit_to_system'],&Help,&Print,
                [&Emphasis_Area,'Artificial Intelligence','Tactical Comp Systems','Software Engineering',
                   'Military Data Processing']],select).
set_title(?wmenu,'QUERY EMPASIS AREA').  disable_menu_item(?m1,[&Print]).
```

```
show_window(?wmenu).
logo().

topic logo.
wh is window(,22,12,46,4,,[Dialogwindow,visible]).
   text('#n #fred                    Loading Data.....').
   use_font(system_font).
   make_modal(?wh). show_window(?wh).

    eof is number_to_char(26).
    message is read_line('track.txt',4).
   repeat
     courseField gets ([?message]) and
     message is read_line('track.txt',4)
   until ?message is ?eof.  close_all(). close_window().
 text(#e).
end.

topic select(item).

   do (?item).

topic 'Artificial Intelligence'.
 message is element(?courseField,1).
 Update().
end.

topic 'Tactical Comp Systems'.
 message is element(?courseField,2).
 Update().
end.
topic 'Software Engineering'.
 message is element(?courseField,3).
 Update().
end.
topic 'Military Data Processing'.
 message is element(?courseField,4).
 Update().
end.


topic update.
   msg is ?message.
   ed1 is first(?message).  message is rest(?message).
   ed2 is first(?message).  message is rest(?message).
   ed3 is first(?message).  message is rest(?message).
   ed4 is first(?message).  message is rest(?message).
  set_title(?wmenu,'QUERY EMPHASIS AREA').
    FormatDisplay().
   button (Continue,OK,18,14,15).

topic ok.
 text(#e). disable_menu_item(?m1,[&Print]).
```

```
      enable_menu_item(?m1,[&Emphasis_Area]).
 end.
end.

topic  FormatDisplay.
   disable_menu_item(?m1,[&Emphasis_Area]).
   enable_menu_item(?m1,[&Print]).
  (*  disable_window(?wmenu).
    wh is window(,6,8,80,17,,[Dialogwindow,visible]).*)
    text('#e #fBLUE

     Emphasis No :


     Track name    :


     Description    :#D ').
      ed1 is edit_line(?ed1,,18,2,3).
      ed2 is edit_line(?ed2,,18,5,30).
      ed3 is edit_line(?ed3,,18,8,40).
      ed4 is edit_line(?ed4,,18,10,40).
end.

 topic &Help.
   wh is window(,11,8,68,16,,[Dialogwindow,visible]).
    text (#e,read ('info.hyp', concat ('//',quemphas) ,'//') ).
    set_file_pos ('info.hyp',0,beginning). make_modal(?wh).
    b2 is button(Ok,continue,30,14).
    topic continue.
       text(#e).
       close('info.hyp'). close_window().
 end.
end.
topic 'Previous_menu'.
     new_file('tempo.dat'). write('tempo.dat',1).
    close_all(). close_window().
    do(!main).
end.
 topic 'quit_to_system'.
      close_all(). close_window().
   clear().
 end.
end.
(*----------END QUEMPHAS.KB----------*)


(*  Program  : Quelec.kb
    Author   : Suprapto
*)

wmenu is window(select:&quit,8,3,72,25,,[Popup,,ControlMenu,ThickFrame,ShowChildren,Siblings]).
set_title(?wmenu,'QUERY ELECTIVE COURSE ').
```

```
m1 is menu([[E&xit,'Previous_Menu','quit_to_system'],&Help,&Print,[&Emphasis_Area,'Artificial Intelligence','Tactical
Comp Systems','Software Engineering',
                    'Military Data Processing']],select).
trackName is ['Artificial Intelligence','Tactical Comp Systems','Software Engineering',
                    'Military Data Processing']. trackNo is [1,2,3,4].
show_window(?wmenu).
QtrName is [Winter,Spring,Summer,Fall]. QtrNo is [1,2,3,4].
wh is window(,22,12,46,4,,[Dialogwindow,visible]).
   text('#n #fred                    Loading Data.....').
   use_font(system_font). disable_menu_item(?m1,[&Print]).
   make_modal(?wh). show_window(?wh).

   trackel is read('TRACKELT.TXT').

   eof is number_to_char(26).
   qtrNumber is []. qtrYear is []. qtrList is [].
   message is read_line('quarter.txt',1).
   repeat
            message is string_to_list(?message,'//') and
            qtrList gets list_to_string(rest(?message),' ')  and
            qtrNumber gets element(string_to_list(first(?message)),2) and
            qtrYear gets element(string_to_list(first(?message)),1) and
            message is read_line('quarter.txt',1)
    until ?message is ?eof. close_all().


 course1 is read('course1.txt').
 courseName is string_to_list(element(string_to_list(?course1,'//'),2),'@').
 course1 is string_to_list(element(string_to_list(?course1,'//'),1),).
 course2 is read('course.txt').

(*  read file course.txt (CourseNumber, CourseName,
                creditLecture,creditLab,{prerequisite},description),
   courseNumbers will be kept in list COURSE1 and
   creditLec, creditLab, prerequisite will be kept in COURSE2
*)


 close_window().

(*  read file course.txt (CourseNumber, CourseName,
                creditLecture,creditLab,{prerequisite},description),
   courseNumbers will be kept in list COURSE1 and
   courseName, creditLec, creditLab, prerequisite will be kept in COURSE2
*)

 topic select(item).

    do (?item).
end.

topic searching. text(#e).
  wh is window,,22,12,46,2,,[Dialogwindow,visible]).
```

```
   text('#fblue                Serching data .....').
    use_font(system_font).
    make_modal(?wh). show_window(?wh).
end.

topic 'Artificial Intelligence'.
  (* searching(). *)
  ed2 is 'Artificial Intelligence'.
   firstfile is string_to_list(element(string_to_list(element(?trackel,1),'//'),1)).
   secondfile  is string_to_list(element(string_to_list(element(?trackel,1),'//'),2)).
 &display_record().
end.

topic 'Tactical Comp Systems'.
 (*   searching(). *)
  ed2 is 'Tactical Computer Systems'.
   firstfile is string_to_list(element(string_to_list(element(?trackel,2),'//'),1)).
   secondfile  is string_to_list(element(string_to_list(element(?trackel,2),'//'),2)).
  &display_record().
end.

topic 'Software Engineering'.
  (* searching(). *)
  ed2 is 'Software Engineering'.
   firstfile is string_to_list(element(string_to_list(element(?trackel,3),'//'),1)).
   secondfile  is string_to_list(element(string_to_list(element(?trackel,3),'//'),2)).
  &display_record().
end.

topic 'Military Data Processing'.
  (* searching(). *) text(#e).
  ed2 is 'Military Data Processing'.
   firstfile is string_to_list(element(string_to_list(element(?trackel,4),'//'),1)).
   secondfile  is string_to_list(element(string_to_list(element(?trackel,4),'//'),2)).
  &display_record().
end.

topic &display_record.
   disable_menu_item(?m1,[&Emphasis_Area]).
   enable_menu_item(?m1,[&Print]).
   text('#e #fblue
-------------------------------------------------------------------------------------------------------
                          EMPHASIS ELECTIVE COURSE

                      TRACK NAME :

  COURSE NO            COURSE NAME                          ELECTIVE TYPE
-------------------------------------------------------------------------------------------------------#d ').
set_display_pos(30,5). text(?ed2).

row is 8.
cp is '#fred Required'. recordNo is list_length(?firstfile). no is 1.
 repeat
```

```
      No = ?no +1 and
      cn is concat('#m','#fred ',element(?firstfile,?no),'#m') and
      cm is concat('#m','#fred ',element(?courseName, where(?course1, element(?firstfile,?no))),'#m')
      and display()
    until ?No is ?recordNo.
    cp is '#fblue Optional'. recordNo is list_length(?secondfile). no is 1.
    repeat
      No = ?no +1 and
      cn is concat('#m','#fblue ',element(?secondfile,?No),'#m') and
      cm is concat('#m','#fblue ',element(?courseName, where(?course1, element(?secondfile,?no))),'#m')
      and display()
    until ?No is ?recordNo. set_display_pos(3,20).
     text('#fblue Please click Course Number, for more information !').
    button(continue,continue,60,20,10).
    topic continue.
      text(#e). enable_menu_item(?m1,[&Emphasis_Area]).
      disable_menu_item(?m1,[&Print]).
     end.
end.
   topic mark(item).
       wh is window(,15,6,60,18,,[Dialogwindow,visible]). n is 2.
      dummy is list_of_char(?item).
      if list_length(?dummy) > 6 then item is element(?course1, where(?courseName, ?item)).
     message is string_to_list(element(?course2, where(?course1, ?item)),'//').
     text('#n#fblue   Course Number  '). set_display_pos(17,?n). text(': '). text(?item).
     text(#n). n = ?n + 2.
     text('#n#fblue   Course Name     '). set_display_pos(17,?n). text(': '). n = ?n + 2.
     text(element(?courseName, where(?course1,?item))). text(#n).
     text('#n#fblue   Credit Lec/Lab     '). set_display_pos(17,?n). text(': '). n = ?n + 2.
     text(first(string_to_list(element(?message,2)))). text(' / ').
     text(last(string_to_list(element(?message,2)))). text(#n).
     text('#n#fblue    Prerequisite       '). set_display_pos(17,?n). text(': '). n = ?n + 2.
     text(element(?message,3)). text(#n).
     text('#n#fblue    Crs description  '). set_display_pos(17,?n). text(': ').
      text(element(?message,4)). text(#n).
     text('                                  '). text(element(?message,5)). text(#n).
     text('#n#fblue    Course offered  in  '). n is 1. wait(,0.000001).
      repeat
      ( if one_of([5,10],?n) then text('#n                                        ')) and
      ( if one_of(string_to_list(element(?QtrList,?n),' '), ?item)
          then  text(element(?qtrName, where(?qtrNo, element(?qtrNumber,?n))))
                and text(' ') and  text(element(?qtrYear,?n)) and text('    ') ) and
      n = ?n + 1
     until ?n is 12.
   use_font(system_font).
   make_modal(?wh).  show_window(?wh).
   button(Ok,continue,48,16,10).
  topic continue.
    close_window().
   end.
end.

topic display.
```

```
   row is ?row+1.
   set_display_pos(5,?row). text(?cn).
  set_display_pos(18,?row). text(?cm).
  set_display_pos(55,?row). text(?cp). wait(,0.00001).
end.

topic can.
    close_window(?wh).
    &menu_option().
end.
topic 'Previous_menu'.
    new_file('tempo.dat'). write('tempo.dat',1).
   close_all(). close_window().
   do(!main).
end.
topic 'quit_to_system'.
     close_all(). close_window().
  clear().
end.

  topic &Print.
     print (get_text (?wmenu)).
   end. (*&Print*)


 topic &Help.
   wh is window(,11,8,66,16,,[Dialogwindow,visible]).
    text (#e,read ('info.hyp', concat ('//',quelec) ,'//') ).
    set_file_pos ('info.hyp',0,beginning). make_modal(?wh).
    b2 is button(Ok,continue,30,14).
    topic continue.
      close('info.hyp'). close_window().
     end.
 end.
(*-------------END QUELEC.KB-----------*)


(* Program : optadec.kb *)

wmenu is window(select:&quit,5,3,82,25,,[Popup,,ControlMenu,ThickFrame,ShowChildren,
                                                    Siblings]).
set_title(?wmenu,'ELECTIVE COURSE SELECTION').
m1  is menu([[E&xit,'Previous_Menu','Quit_to_System'],&Help,&Print ,
                &Student],select).
trackList is ['Artificial Intelligence','Tactical Comp Systems','Software Engineering',
          'Military Data Processing'].
QtrNum is [1,2,3,4,5,6,7,8,9,10,11,12].
QtrYear is ['91 1','91 2','91 3','91 4','92 1','92 2','92 3','92 4','93 1','93 2','93 3','93 4'].
Year is ['Win 91','Spr 91','Sum 91','Fall 91','Win 92','Spr 92','Sum 92','Fall 92',
          'Win 93','Spr 93','Sum 93','Fall 93'].
QtrName is [Winter,Spring,Summer,Fall].
requiredCourse is [cs2970,cs3200,cs3300,cs3310,cs3320,cs3111,cs3450,
                          cs3460].
```

```
disable_menu_item(?m1,[&Print]).


show_window(?wmenu).

wh is window(,22,14,46,4,,[Dialogwindow,visible]).
   text('#n #fred                    Loading Data.....').
   use_font(system_font).
   make_modal(?wh). show_window(?wh).

read_file(). close_window().

topic read_file.
quarter is read('QUARTER.TXT').
(* read file quarter.txt (quarter#, year, {course#})
  It contains courses to be offered in each quarter *)

 course1 is read('course1.txt').
 courseName is string_to_list(element(string_to_list(?course1,'//'),2),'@').
 course1 is string_to_list(element(string_to_list(?course1,'//'),1),).
 course2 is read('course.txt').

(*  read file course.txt (CourseNumber, CourseName,
                creditLecture,creditLab,{prerequisite},description),
   courseNumbers will be kept in list COURSE1 and
   creditLec, creditLab, prerequisite will be kept in COURSE2
*)


trackel is read('TRACKELT.TXT').
(* read track required/elective courses  *)

SList is read('student1.txt').
SInfo is read('std.txt').
(* read file student.txt that contains student_Id, name (kept in SList) and
            student Information(name,rank,section etc) kept in SInfo  *)

 text(#e).   track_name is ' '.  sid is []. snm is ' '.
end.

 topic select(item).
    do (?item).
 end.

topic &Student.
        wh is window(,15,10,60,12,,[Dialogwindow,visible]).
        use_font(system_font).
        make_modal(?wh).
       set_display_pos(8,2).
       text('#fblue Select Student Id and  name from the List : ').
       list_box([?SList],input_student,8,4,32,5,,t,list_select_event).
       button(Ok,commit,45,4,10).
       button(Cancel,can,45,7,10).
```

```
end.

topic can. close_window(). cancel(). end.

topic input_student(item).
  set_display_pos(27,10). text('                    ').
  set_display_pos(8,10). text('#fblue Student_id & Name : '). text(?item).
  SKey is ?item.
end.

topic commit.
  close_window().
  sid is string_to_list(?skey).  snm is concat(element(?sid,2),' ',element(?sid,3)).
  sid is first(?sid).
  trackNo is string_to_list(element(?SInfo, where(?SList,?Skey)),'//').
  ssc is element(?trackNo,3). srk is element(?trackNo,2).
  strqtr is element(?trackNo,5).
  endqtr is string_to_list(element(?trackNo,6)).
  SltCrs is string_to_list(element(?trackNo,7),'@').
  TknCrs is string_to_list(element(?trackNo,8)).
  trackNo is element(?trackNo,4).
  required is string_to_list(element(?trackel,?trackNo),'//').
  elective is element(?required,2).
   required is element(?required,1).
  track_name is element(?trackList,?trackNo).
  dialogBox().
(* searchs the information for selected student
    such as student_id, name, track, selected courses, and
    courses that has already been taken *)
end.

topic dialogBox.
  (* displays the selection *) text(#e).
   set_display_pos(23,7) and text('#fblue Track          : ').
   text(?track_name).
   set_display_pos(23,9).
   text('#fblue Student-Id : '). text(?sid).
   set_display_pos(23,11). text('#fblue N a m e      : '). text(?snm).
   button(Ok,SEARCHING,23,14,10).
   button(Cancel,cancel,40,14,10).
end.

topic modify. close_window(). end.
topic postpone. close_window(). cancel(). end.

topic searching.
 (* searchs the data associate with the selected student
  from fourth quarter up to 8th  *)


  Nu is element(?QtrNum, where(?QtrYear,?StrQtr)).
  SltCrs is string_to_list(?SltCrs,'@').
  Qtr4 is string_to_list(element(?SltCrs,1)).
```

```
  Qtr5 is string_to_list(element(?SltCrs,2)).
  Qtr6 is string_to_list(element(?SltCrs,3)).
  Qtr7 is string_to_list(element(?SltCrs,4)).
  Qtr8 is string_to_list(element(?SltCrs,5)).

  Fourth is string_to_list(element(string_to_list(element(?Quarter,?Nu),'//'),2)).
  Fifth is string_to_list(element(string_to_list(element(?Quarter,?Nu+1),'//'),2)).
  Sixth is string_to_list(element(string_to_list(element(?Quarter,?Nu+2),'//'),2)).
  Seventh is string_to_list(element(string_to_list(element(?Quarter,?Nu+3),'//'),2)).
  Eighth is string_to_list(element(string_to_list(element(?Quarter,?Nu+4),'//'),2)).
  display_header().
  process(4,element(?year,where(?qtrnum,?Nu)),13,?Qtr4,?Fourth).
  process(5,element(?year,where(?qtrnum,?Nu+1)),14,?Qtr5,?Fifth).
  process(6,element(?year,where(?qtrnum,?Nu+2)),15,?Qtr6,?Sixth).
  process(7,element(?year,where(?qtrnum,?Nu+3)),16,?Qtr7,?seventh).
  process(8,element(?year,where(?qtrnum,?Nu+4)),17,?qtr8,?Eighth).
  text(#n). text('                    #fgray N #fblue = the course was not offered ').
  text(#n). text('                    #fgray P #fblue = prerequisite course should be taken first').
  button(Continue,cancel,72,21,9).
end.
topic process(Q,Y,L,courses,quarter).
    recordNo is list_length(?courses). no is 1.
    set_display_pos(2,?L). text(?Q). text(' / ').text(?Y).  R is 14.
    repeat

        cn is element(?courses,?no) and process1(?R) and r is ?r + 12 and
        no = ?no +1
    until ?no > ?recordNo.
set_display_pos(1,18).
text('#fblue
 ----------------------------------------------------------------------------------------------------------------------------------#d ').

 topic process1.
    getdata is string_to_list(element(?course2, where(?course1, ?cn)),'//').
    prereq is element(?getdata,3).
    if not(?prereq is ' ') then prereq is string_to_list(?prereq).
    checkprereq is different(?prereq,?tknCrs).
    if ?checkprereq is [] or ?checkprereq is [' '] then p is ' ' else p is '#fgray P'.
    if not(intersect(?cn,?quarter) is []) then n is ' ' else n is '#fgray N'.
    tkncrs is combine(?tknCrs,?cn).
    set_display_pos(?R,?L).
    if ?p is '#fgray P' or ?n is '#fgray N' then text(concat('#m','#fgray ',?cn,'#m'))
                    else text(concat('#m',?cn,'#m')).
    text(' '). text(?n). text(?p). wait(,0.0001).
  end.
end.



topic display_header.
  disable_menu_item(?m1,[&Student]). enable_menu_item(?m1,[&Print]).
     text('#e #fblue
```

COMPUTER SCIENCE (368)
OPTION AREA DECLARATION AND COURSE SELECTION

EMPHASIS AREA
STUDENT-ID
NAME
RANK
SECTION

QUARTER COURSE COURSE COURSE COURSE COURSE COURSE

---#d ').

```
set_display_pos(34,5). text(':'). set_display_pos(36,5). text(?track_name).
set_display_pos(34,6). text(':'). set_display_pos(36,6). text(?sid).
set_display_pos(34,7). text(':'). set_display_pos(36,7). text(?snm).
set_display_pos(34,8). text(':'). set_display_pos(36,8). text(?srk).
SET_DISPLAY_POS(34,9). TEXT(':'). set_display_pos(36,9). text(?ssc).
row is 12.
end.


topic cancel.
  text(#e). track_name is ' '. sid is ' '. snm is ' '.
  disable_menu_item(?m1,[&Print]). enable_menu_item(?m1,[&Student]).
end.
   topic mark(item).
       wh is window(,15,6,60,15,,[Dialogwindow,visible]). n is 2.
      dummy is list_of_char(?item).
      if list_length(?dummy) > 6 then item is element(?course1, where(?courseName, ?item)).
     message is string_to_list(element(?course2, where(?course1, ?item)),'//').
     text('#n#fblue   Course Number  '). set_display_pos(17,?n). text(': '). text(?item).
     text(#n). n = ?n + 2.
     text('#n#fblue   Course Name     '). set_display_pos(17,?n). text(': '). n = ?n + 2.
     text(element(?courseName, where(?course1,?item))). text(#n).
     text('#n#fblue   Credit Lec/Lab     '). set_display_pos(17,?n). text(': '). n = ?n + 2.
     text(first(string_to_list(element(?message,2)))). text(' / ').
     text(last(string_to_list(element(?message,2)))). text(#n).
     text('#n#fblue   Prerequisite       '). set_display_pos(17,?n). text(': '). n = ?n + 2.
     text(element(?message,3)). text(#n).
     text('#n#fblue   Crs description  '). set_display_pos(17,?n). text(': ').
      text(element(?message,4)). text(#n).
     text('                                  '). text(element(?message,5)). text(#n).

    use_font(system_font).
    make_modal(?wh). show_window(?wh).
    button(Ok,continue,48,13,10).
   topic continue.
     close_window().
    end.
end.
topic 'Previous_menu'.
  new_file('tempo.dat'). write('tempo.dat',1).
  close_all().
  close_window().
```

```
 do(!main).
end.

topic 'quit_to_system'.
 close_all(). close_window().
 clear().
end.

topic &Print.
     print (get_text (?wmenu)).
end. (*&Print*)


topic &Help.
  wh is window(,11,8,69,17,,[Dialogwindow,visible]).
  text (#e,read ('info.hyp', concat ('//',optadec) ,'//') ).
  set_file_pos ('info.hyp',0,beginning).
  b2 is button(Ok,continue,30,15).
  set_focus(?b2).
  wait().
  close('info.hyp'). close_window().
end.
(*----------END OPTADEC.KB----------*)



(*  Program     : offered.kb
    Author      : suprapto
*)
 wmenu is window(select:&quit,8,2,72,28,,[Popup,,ControlMenu,ThickFrame,ShowChildren,Siblings]).
set_title(?wmenu,'ELECTIVE COURSE OFFERED').
m                                                    1      i                                                    s
menu([[E&xit,'Previous_Menu','Quit_to_system'],&Help,&Print,[&Quarter_Name,Winter,Spring,Summer,Fall],[&Year
,'91','92','93']],select).
disable_menu_item(?m1,[&Print]).
show_window(?wmenu). q is 0.  y is 0.

wh is window(,22,14,46,4,,[Dialogwindow,visible]).
   text('#n #fred                    Loading Data.....').
   use_font(system_font).
   make_modal(?wh). show_window(?wh).
read_file(). close_window().

topic read_file.
quarter is read('quarter.txt').
  course1 is read('course1.txt').
  courseName is string_to_list(element(string_to_list(?course1,'//'),2),'@').
  course1 is string_to_list(element(string_to_list(?course1,'//'),1)).
  course2 is read('course.txt').

(*  read file course.txt (CourseNumber, CourseName,
                creditLecture,creditLab,{prerequisite},description),
    courseNumbers will be kept in list COURSE1 and
```

```
   creditLec, creditLab, prerequisite will be kept in COURSE2
*)

  text(#e).
end.

 topic select(item).

     do (?item).
 end.
topic '91'.  year is '91'.  y is 1. dialogBox(). end.
topic '92'.  year is '92'.  y is 1. dialogBox().  end.
topic '93'.  year is '93'.  y is 1. dialogBox(). end.

topic 'Fall'.   quarters is 'Fall'. qtr is 4.  q is 1. dialogBox(). end.
topic 'Winter'.  quarters is 'Winter'. qtr is 1.  q is 1. dialogBox(). end.
topic 'Spring'.   quarters is 'Spring'.  qtr is 2. q is 1. dialogBox(). end.
topic 'Summer'.  quarters is 'Summer'.  qtr is 3.  q is 1. dialogBox(). end.

topic dialogBox.
  if ?q is 1 then
       ( if ?y is 1 then text(#e) and set_display_pos(28,6) and text('#fred Y o u r   c h o i c e') and
                         set_display_pos(28,8) and text('#fblue Quarter : ') and text(?quarters) and
                         set_display_pos(28,10) and text('#Fblue Y e a r : ') and text(?year) and
                          button(Ok,SEARCHING,28,12,6) and
                          button(Cancel,cancel,38,12,10)
            else  set_display_pos(28,8) and text('Quarter : ') and text(?quarters) and
                         set_display_pos(20,10) and text(' Please select a academic year ! ') )
   else set_display_pos(28,8) and text(' Y e a r : ') and text(?year) and
                         set_display_pos(20,10) and text(' Please select a Quarter Name ! ').
end.

topic searching.
wh is window(,22,14,46,4,,[Dialogwindow,visible]).
   text('#n #fred                    Searching Data.....').
   use_font(system_font).
   make_modal(?wh). show_window(?wh).
 recordNo is list_length(?quarter). no is 1. header is 0. quarterList is []. data is [].
 repeat
    cn is element(?quarter,?no) and
    cn = string_to_list(?cn,'//') and
    (  if ( element(?cn,1) is concat(?year,' ',?qtr)) then
            quarterList is string_to_list(rest(?cn),' ') and no is ?recordNo and
            quarterList is intersect(?quarterList,?course1)) and
       no is ?no + 1 and data gets element(?cn,1)
     until ?no > ?recordNo.
    if ?quarterList is [] then data is list_to_string(?data,' ') and
       message is '1-winter 2-spring'
     else
         displayRecord().
end.
topic displayRecord.
   display_header().
```

```
   recordNo is list_length(?quarterList). no is 1.
   repeat
      getdata is string_to_list(element(?course2, where(?course1, element(?quarterList,?no))),'//')
      and  cm is element(?courseName, where(?course1, element(?quarterList, ?no))) and
      Lec is element(?getdata,2) and Lec is string_to_list(?lec) and
      Lab is element(?Lec,2) and Lec is element(?Lec,1) and display()  and
      No = ?no +1
 until ?No > ?recordNo. enable_menu_item(?m1,[&Print]).
 disable_menu_item(?m1,[E&xit,&Quarter_Name,&Year]).
  text('#n#fblue ----------').
 button(continue,continue,60,22,10).
 topic continue.  text(#e).
    enable_menu_item(?m1,[E&xit,&Quarter_Name,&Year]).
    disable_menu_item(?m1,[&Print]).
 end.
end.


(*-------------------------------------------------------------------*)

topic display_header.
  close_window().
text('#e #fblue
 ---------------------------------------------------------------------------------------------------
                          ELECTIVE COURSE OFFERED
                   QUARTER NAME :                    YEAR :
 COURSE NO        COURSE NAME              CREDIT LEC/LAB
 ---------------------------------------------------------------------------------------------------#d ').
set_display_pos(33,4). text(?quarters). set_display_pos(56,4). text(?year).
row is 6.
end.


topic display.
   row is ?row+1.
   set_display_pos(5,?row). text(element(?quarterList,?no)).
  set_display_pos(15,?row). text(?cm).
  set_display_pos(50,?row). text(?Lec).
  set_display_pos(52,?row). text('/'). set_display_pos(54,?row). text(?Lab).
   wait(,0.0000001).
end.

topic cancel.
 text(#e). y is 0. q is 0.
end.
topic 'Previous_menu'.
    new_file('tempo.dat'). write('tempo.dat',1).
   close_all(). close_window().
   do(!main).
end.
topic 'quit_to_system'.
     close_all(). close_window().
  clear().
```

```
end.

  topic &Print.
     print (get_text (?wmenu)).
   end. (*&Print*)


 topic &Help.
   wh is window(,11,8,64,17,,[Dialogwindow,visible]).
    text (#e,read ('info.hyp', concat ('//',quequart) ,'//') ).
    set_file_pos ('info.hyp',0,beginning). make_modal(?wh).
    b2 is button(Ok,continue,30,15).
    set_focus(?b2).
    wait().
    close('info.hyp'). close_window().
 end.
(*----------END OFFERED.KB----------*)


(* Program : genselec.kb *)

wmenu is window(select:&quit,5,3,82,25,,[Popup,,ControlMenu,ThickFrame,ShowChildren,
                                                    Siblings]).
set_title(?wmenu,'ELECTIVE COURSE SELECTION').
m1  is menu([[E&xit,'Previous_Menu','Quit_to_System'],&Help,&Print ,
                  [&Quarter,Winter,Spring,Summer,Fall],
                  [&Year,'91','92','93'],&Student],select).
trackList is ['Artificial Intelligence','Tactical Comp Systems','Software Engineering',
           'Military Data Processing'].
QtrName is [Winter,Spring,Summer,Fall].
requiredCourse is [cs2970,cs3200,cs3310,cs3320,cs3111,cs3450,cs3460,cs3502,
                          cs3601,cs3650,cs4601].
disable_menu_item(?m1,[&Print]).


show_window(?wmenu).

wh is window(,22,14,46,4,,[Dialogwindow,visible]).
   text('#n #fred                    Loading Data.....').
   use_font(system_font).
   make_modal(?wh). show_window(?wh).

read_file(). close_window().

topic read_file.
quarter is read('QUARTER.TXT').
(* read file quarter.txt (quarter#, year, {course#})
  It contains courses to be offered in each quarter *)

 course1 is read('course1.txt').
 courseName is string_to_list(element(string_to_list(?course1,'//'),2),'@').
 course1 is string_to_list(element(string_to_list(?course1,'//'),1),).
 course2 is read('course.txt').
```

```
(*   read file course.txt (CourseNumber, CourseName,
                    creditLecture,creditLab,{prerequisite},description),
    courseNumbers will be kept in list COURSE1 and
    creditLec, creditLab, prerequisite will be kept in COURSE2
*)


trackel is read('TRACKELT.TXT').
(* read track required/elective courses  *)

SList is read('student1.txt').
SInfo is read('student.txt').
(* read file student.txt that contains student_Id, name (kept in SList) and
              student Information(name,rank,section etc) kept in SInfo  *)


 text(#e).  year is ' '.  quarters is ' '.  track_name is ' '.  sid is []. snm is ' '.
end.

 topic select(item).

     do (?item).
 end.

topic &Student.
          wh is window(,15,10,60,12,,[Dialogwindow,visible]).
          use_font(system_font).
          make_modal(?wh).
         set_display_pos(8,2).
         text('#fblue Select Student Id and  name from the List : ').
         list_box([?SList],input_student,8,4,32,5,,t,list_select_event).
         button(Ok,commit,45,4,10).
         button(Cancel,can,45,7,10).
end.

topic can. close_window(). cancel(). end.

topic input_student(item).
  set_display_pos(27,10). text('                                    ').
  set_display_pos(8,10). text('#fblue Student_id & Name : '). text(?item).
  SKey is ?item.
end.

topic commit.
  close_window().
  sid is string_to_list(?skey).  snm is concat(element(?sid,2),' ',element(?sid,3)).
  sid is first(?sid).
  trackNo is string_to_list(element(?SInfo, where(?SList,?Skey)),'//').
  ssc is element(?trackNo,3). srk is element(?trackNo,2).
  strqtr is string_to_list(element(?trackNo,5)).
  endqtr is string_to_list(element(?trackNo,6)).
  SltCrs is string_to_list(element(?trackNo,7)).
```

```
  TknCrs is string_to_list(element(?trackNo,8)).
  trackNo is element(?trackNo,4).
  required is string_to_list(element(?trackel,?trackNo),'//').
  elective is element(?required,2).
   required is element(?required,1).
  track_name is element(?trackList,?trackNo).
  dialogBox().
(* searchs the information for selected student
    such as student_id, name, track, selected courses, and
    courses that has already been taken *)
end.

topic '91'.  year is '91'.  dialogBox(). end.
topic '92'.  year is '92'.  dialogBox(). end.
topic '93'.  year is '93'.  dialogBox(). end.


topic 'Fall'.    quarters is 'Fall'.  q is '4'.  dialogBox(). end.
topic 'Winter'.  quarters is 'Winter'.  q is '1'. dialogBox(). end.
topic 'Spring'.  quarters is 'Spring'. q is '2'.  dialogBox(). end.
topic 'Summer'.  quarters is 'Summer'.  q is '3'.  dialogBox(). end.



topic dialogBox.
  (* displays the selection *) text(#e).
   set_display_pos(23,4) and text('#fblue Track          : ').
   text(?track_name).
   set_display_pos(23,6) and text('#fblue Quarter       : ').
   text(?quarters). text('     ').
   set_display_pos(23,8) and text('#Fblue Y e a r       : ').
   text(?year).
   set_display_pos(23,10).
   text('#fblue Student-Id : '). text(?sid).
   set_display_pos(23,12). text('#fblue N a m e     : '). text(?snm).

 (* check the student whether he/she is in the selected quarter or not *)

   if (not(?year is ' ') and not(?Quarters is ' ') and not(?snm is ' ')) then
    ( if (concat(?year,?q)) < list_to_string(?endqtr) + 1 and
         (concat(?year,?q)) >  list_to_string(?strqtr) - 1 then
            button(Ok,SEARCHING,23,14,10) and
            button(Cancel,cancel,40,14,10)
      else     wh is window(,22,19,48,6,,[Dialogwindow,visible]) and
    text('#fred       The quarter does not fix to the student.....') and
    text('#n #fblue  Please select  from ') and text(element(?QtrName,element(?strqtr,2))) and
    text(' ') and
    text(element(?strqtr,1)) and text('#fblue  through ') and
    text(element(?QtrName,element(?endqtr,2))) and text(' ') and
    text(element(?endqtr,1)) and
    use_font(system_font) and
    make_modal(?wh) and button(Modify,modify,10,4,10) and
                                  button(Cancel,postpone,30,4,10)).
```

```
end.

topic modify. close_window(). end.
topic postpone. close_window(). cancel(). end.

topic searching.
 (* searchs the data associate with the selected student
  track courses, selected courses, courses that has already
  been taken, courses has not been taken, courses offered in
  the selected quarter *)

  recordNo is list_length(?quarter). no is 1. header is 0. quarterList is [].

    (* combine required and elective course into topic ELECTIVE *)
    elective is combine(string_to_list(?elective),string_to_list(?required)).

  (* search quarter offering courses for selected quarter *)
  repeat
     cn is element(?quarter,?no) and
     cn = string_to_list(?cn,'//') and
     ( if element(?cn,1) is concat(?year,' ',?q) then
                 cn is string_to_list(rest(?cn),' ') and
               quarterList is ?cn  and
                no is ?recordNo -1) and
      No = ?no +1
    until ?No is ?recordNo.
    if ?quarterList is [] then wh is window(,22,11,46,2,,[Dialogwindow,visible]) and
                                    text('#fred                    Data Not Found') and
                                    wait(,0.0001) and close_window() and
                                    enable_menu_item(?m1,[&Year,&Quarter,&Student])
       else yesDisplay().
end.

topic yesDisplay.
(* displays the student information and course selection *)
     text(#e).
     set_display_pos(20,2). text('#fblue Emphasis_Area  : '). text(?track_name).
     set_display_pos(20,3). text('#fblue Quarter / Year     : ').
     text(?quarters). text(' '). text(?year).
     set_display_pos(20,4). text('#fblue  Student_Id         : '). text(?sid).
     set_display_pos(20,5). text('#fblue  Student_name   : '). text(?snm).
     set_display_pos(1,8). courses is [].
     text('#fblue Track Courses    Selected Courses   Have been taken   Not been taken #fred Courses Offered').
     list_box([combine(?requiredCourse,?elective)],,3,9,10,,t,t).
     list_box([combine(?requiredcourse,?sltCrs)],,19,9,10,,t,t).
     list_box([?tknCrs],,36,9,10,,t,t).
     notTkn is different(combine(?requiredCourse,?sltCrs), ?tknCrs).
     dif is different(?quarterList,?tknCrs).
     list_box([?notTkn],,53,9,10,,t,t).
     list_box([intersect(?notTkn,?dif)],input_data,70,9,10,,t,t,list_select_event).
     set_display_pos(25,18). text('#fblue Please select courses to be taken').
     set_display_pos(25,19). text('#fblue from "#fred Courses Offered" #fblue list_box !!').
```

```
    button(Ok,lookup,70,17,10).
    button(Cancel,cancel,70,20,10).
   disable_menu_item(?m1,[&Year,&Quarter,&Student]).
end.
topic input_data(item).  courses is ?item. end.
topic displayInfo.
   wh is window(,22,11,48,7,,[Dialogwindow,visible]).
   text('#n #fred  Please select available courses offered.....').
   use_font(system_font).
   make_modal(?wh).  button('Try again',modify,10,4,12). button(Cancel,postpone,30,4,10).
end.

topic lookup.
    if ?courses is []  then displayInfo()
       else searchData().
end.
topic SearchData.

 wh is window(,22,15,46,4,,[Dialogwindow,visible]).
   text('#n #fblue                    Searching data .....').
   use_font(system_font).
   make_modal(?wh). show_window(?wh).
   recordNo is list_length(?courses). no is 1. header is 0.
   repeat
     getdata is string_to_list(element(?course2, where(?course1, element(?courses,?no))),'//')
      and  cm is element(?courseName, where(?course1, element(?courses, ?no))) and
           Lec is element(?getdata,2) and Lec is string_to_list(?lec) and
           Lab is element(?Lec,2) and Lec is element(?Lec,1) and
           prereq is element(?getdata,3) and
           ( if not(?prereq is ' ') then prereq is string_to_list(?prereq)) and
           display() and
      no = ?no + 1
     until ?no > ? recordNo.
    button(Continue,continue,72,21,9).

  (*  if checkPrereq is [] then
    new_taken is concat(?tkn_Crs,' ',list_to_string(?courses,' ')) and
     takenCrs is replace(?takenCrs, element(?takenCrs, where(?scn,?sid)),?new_taken).
  *)
  topic continue.
     enable_menu_item(?m1,[&Year,&Quarter,&Student]). cancel().
   end.
end.

(*-------------------------------------------------------------------*)

topic display_header. close_window().
    text('#e #fblue
                          ELECTIVE COURSE SELECTION
                          ------------------------------------------------

                    EMPHASIS  AREA
                    QUARTER / YEAR
```

```
                        STUDENT-ID
                             NAME
                             RANK
                          SECTION

   COURSE NO          COURSE NAME          CREDIT LEC/LAB      REMARK
   ---------------------------------------------------------------------------------------------------------------#d ').
set_display_pos(34,5). text(':'). set_display_pos(36,5). text(?track_name).
set_display_pos(34,6). text(':'). set_display_pos(36,6). text(?quarters).
text(' / ' ).          text(?year).
set_display_pos(34,8). text(':'). set_display_pos(36,8). text(?sid).
set_display_pos(34,9). text(':'). set_display_pos(36,9). text(?snm).
set_display_pos(34,10). text(':'). set_display_pos(36,10). text(?srk).
SET_DISPLAY_POS(34,11). TEXT(':'). set_display_pos(36,11). text(?ssc).
row is 14.
end.


topic display.
  if ?header is 0 then header is 1 and enable_menu_item(?m1,[&print]) and
                                display_header().
  row is ?row+1. checkPrereq is different(?prereq,?tknCrs).
  if  ?checkPrereq is [] or ?checkPrereq is [' ']
              then remark is '            valid ' and display_valid()
   else remark is concat('not valid, prereq ',element(?CheckPrereq,1)) and
         display_invalid().
end.
topic display_valid.
  set_display_pos(5,?row).  text(concat('#m',element(?courses,?no),'#m')).
  set_display_pos(18,?row). text(concat('#m',?cm,'#m')).
  set_display_pos(50,?row). text(?Lec).
  set_display_pos(52,?row). text('/'). set_display_pos(54,?row). text(?Lab).
  set_display_pos(58,?row). text(?remark). wait(,0.0001).
end.
topic display_invalid.
  set_display_pos(5,?row).
   text(concat('#m','#fgray ',element(?courses,?no),'#m')).
  set_display_pos(18,?row). text(concat('#m','#fgray ',?cm,'#m')).
  set_display_pos(50,?row). text(concat('#fgray ',?Lec)).
  set_display_pos(52,?row). text('/'). set_display_pos(54,?row).
                                    text(concat('#fgray ',?Lab)).
  set_display_pos(58,?row). text(concat('#fgray ',?remark)). wait(,0.0001).
end.
topic cancel.
  text(#e).  year is ' '.  quarters is ' '.  track_name is ' '. sid is ' '. snm is ' '.
  disable_menu_item(?m1,[&Print]). enable_menu_item(?m1,[&Year,&Quarter,&Student]).
end.
   topic mark(item).
       wh is window(,15,6,60,15,,[Dialogwindow,visible]). n is 2.
      dummy is list_of_char(?item).
      if list_length(?dummy) > 6 then item is element(?course1, where(?courseName, ?item)).
     message is string_to_list(element(?course2, where(?course1, ?item)),'//').
     text('#n#fblue   Course Number  '). set_display_pos(17,?n). text(': '). text(?item).
```

```
    text(#n). n = ?n + 2.
    text('#n#fblue   Course Name     '). set_display_pos(17,?n). text(': '). n = ?n + 2.
    text(element(?courseName, where(?course1,?item))). text(#n).
    text('#n#fblue   Credit Lec/Lab     '). set_display_pos(17,?n). text(': '). n = ?n + 2.
    text(first(string_to_list(element(?message,2)))). text(' / ').
    text(last(string_to_list(element(?message,2)))). text(#n).
    text('#n#fblue   Prerequisite       '). set_display_pos(17,?n). text(': '). n = ?n + 2.
    text(element(?message,3)). text(#n).
    text('#n#fblue   Crs description  '). set_display_pos(17,?n). text(': ').
     text(element(?message,4)). text(#n).
    text('                                  '). text(element(?message,5)). text(#n).

   use_font(system_font).
   make_modal(?wh).  show_window(?wh).
   button(Ok,continue,48,13,10).
  topic continue.
    close_window().
   end.
end.
topic 'Previous_menu'.
 (* updateFile(). *)
  new_file('tempo.dat'). write('tempo.dat',1).
  close_all().
  close_window().
  do(!main).
end.
topic updateFile.
  length is list_length(?scn). no is 1. courses is [].
  repeat
     courses gets element(?selectedCrs,?no) and
     courses gets element(?takenCrs,?no) and no = ?no +1
   until ?no > ?length.
   new_file('student.txt').  write('student.txt',?courses).
end.

topic 'quit_to_system'.
  (* updateFile(). *)
  close_all().   close_window().
  clear().
end.

topic &Print.
     print (get_text (?wmenu)).
end. (*&Print*)


topic &Help.
  wh is window(,11,8,69,17,,[Dialogwindow,visible]).
  text (#e,read ('info.hyp', concat ('//',quegener) ,'//') ).
  set_file_pos ('info.hyp',0,beginning).
  b2 is button(Ok,continue,30,15).
  set_focus(?b2).
  wait().
```

```
  close('info.hyp'). close_window().
end.
(*----------------END GENSELEC.KB-------------*)
```

LIST OF REFERENCES

Bev & Bill Thompson, *KnowledgePro (Windows)*, Knowledge Garden, Inc., 1990.

Dan Shafer, *Object-Oriented Tools Make it Easier To Program Windowing Applications*, Windows Shopper's Guide, White Fox Communications, 1989.

E.S. Cohen, E.T. Smith, and L.A. Iverson, *Contraint-Based Tiled Windows*, Proc. 1st Int'l Conf. on Computer Workstations, CS Press, Los Alamitos, Calif., Nov. 1984, pp. 2-11.

Foley., van Dam., Feiner., *Computer Graphics : Principles and Practice*, Addison-Wesley Publishing Company Inc. 1987.

Herot, C. F. *Spatial management of data*, ACM Transactions on Database Systems, Vol 5, No 4 (Dec. 1980), pp. 493-514.

Hirakawa, Tanaka, Ichikawa, *HI-VISUAL: A Language for Supporting Visual Interaction in Programming*, Proceeding of the 1984 IEEE Workshop in Visual Language (Dec. 1984), pp. 199-205.

Keister, R. S., and Gallaway, G.R., *Making software user friendly : An assessment of data entry performance*, Proceedings of the Human Factors Society 27th Annual Meeting, pp 1031-1034. Santa Monica, Ca Human Factors Society. 1983.

Larson, J. A. *The forms pattern language*, Proceedings of IEEE International Conference on Data Engineering (Los Angeles, 1984), pp. 183-191.

Mayhew, D., *Principles and Guidelines in User Interface Design*, Printice-Hall, Englewood Cliffs, NJ, 1990.

Morland, V. *Human factor guidelines for terminal interface design*, Communication of the ACM, 1983, pp. 484-494.

Nan C. Shu, *Visual Programming*, Van Nonstrand Reinhold Company Inc., 1988.

Shneiderman B., *Designing the user interface: strategies for effective human-computer interaction*, Reading, MA: Addison-Wesley, 1987.

Tekla S. Perry, and J., Voelcker, *Of mice and menus : designing the user-friendly interface*, IEEE Spectrum, September 1989.

Thomas S. Tullis, *Screen design*, Handbok of Human-Computer Interaction, Elsevier Science Publishers B,V.(North-Holland), 1988.

Tullis, T. S., *An evaluation of alphanumeric, graphic, and color information displays*, Human Factors, 23, 1981, pp. 541-550.

Wu, C. T., *A new graphics user interface for accessing a database*, Proceedings of Computer Graphics Tokyo '86 (Tokyo, 1986), pp. 203-219.

Wu, C. T., *GLAD: Graphics Language for Databases*, Proceedings of 11th International Conference on Computer Software and Applications (Tokyo, 1987).

Wu, C. T. and D. K. Shiao, *Implementation of Visual Database Interface using an Object-Oriented Language*, Naval Postgraduate School Technical Report NPS52-88-050, June 1988.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Department Chairman, Code CS Department of Computer Science Naval Postgradute School Monterey, California 93943-5000	1
4.	Professor C. Thomas Wu, Code CS Wq Naval Postgraduate School Monterey, California 93943-5000	2
5.	CDR Bruce B. Giannotti, Code CS Gg Naval Postgradute School Monterey, California 93943-5000	1
6.	Office of Defense Attache Embassy of the Republic of Indonesia 2020 Massachusetts Avenue, N.W. Washington, D.C., 20036	2
7.	Kadispullahta TNI-AU Jl. Gatot Subroto Jakarta, Indonesia	2
8.	Dan Kodikau Jl. Maj. Sungkono Surabaya, Indonesia	1
9.	Suprapto Jl. Albatros No 7 Surabaya, Indonesia	2